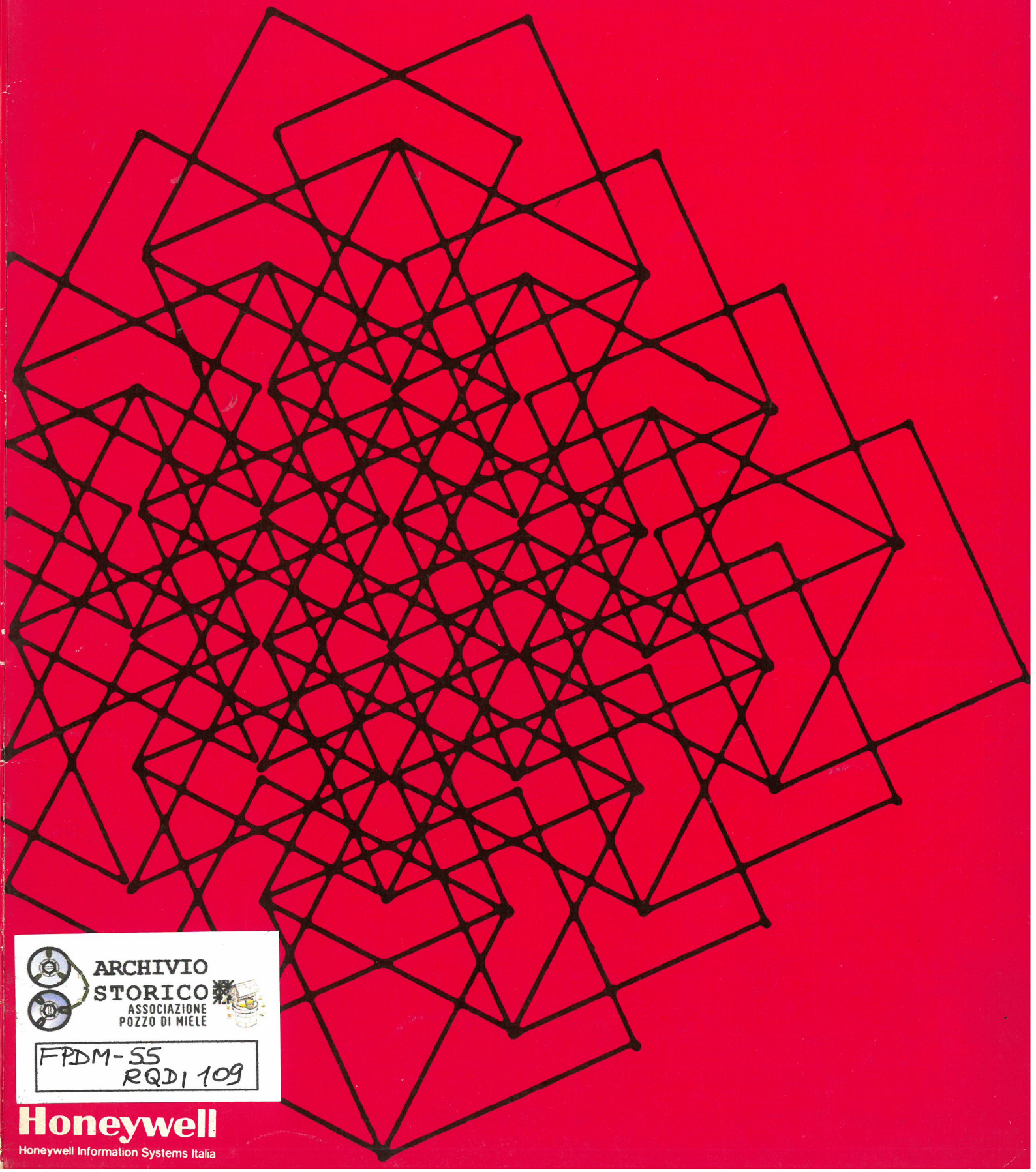
Quaderni di informatica 11

Rassegna di tecnologia ed applicazioni degli elaboratori elettronici - Anno VI - Numero 1 - 1979



ARCHIVIO STORICO
ASSOCIAZIONE POZZO DI MIELE
FPDM-55
RQD1 109

Honeywell
Honeywell Information Systems Italia

FPDM - 55

Quaderni di informatica

Rassegna di tecnologia ed applicazioni degli elaboratori elettronici
Anno VI - Numero 1 - 1979

Sommario

PHOS: una evoluzione nelle metodologie di programmazione.
Negli ultimi anni sono state introdotte nuove idee per razionalizzare il lavoro di programmazione. La metodologia PHOS (Program Hierarchy from Output Structure) costituisce una sintesi dell'evoluzione concettuale del settore, con significativi risvolti sul piano pratico.
M. MAIOCCHI, E. SPOLETINI

La comunicazione uomo-macchina e i linguaggi di programmazione
Se esiste una «barriera del linguaggio» tra gli uomini, a maggior ragione il problema si pone tra l'uomo ed il computer. I termini del problema, gli approcci adottati, le linee di tendenza costituiscono la materia di questo articolo.
G.M. PALERMO, G. DAMAGGIO

La produttività di programmazione nello sviluppo del software
La previsione delle risorse e dei tempi necessari per uno sviluppo software è un'aspetto delicato dell'EDP. In questo articolo vengono forniti criteri per una stima oggettiva, sulla base di indici di produttività effettivamente rilevati.
D.A. WALSH

L'automazione dell'ufficio
Il cosidetto «lavoro d'ufficio» assorbe nella società moderna una quota imponente di risorse. Questo articolo analizza tale insieme di attività e ne prospetta una radicale evoluzione attraverso le tecniche congiunte dell'automazione e della comunicazione.
G. OCCHINI

Animazione grafica mediante «Computer Output Microfilm» (COM)
L'uso del COM consente di sintetizzare graficamente i risultati di laboriose elaborazioni elettroniche, permettendo l'osservazione cinematografica di fenomeni fisici complessi.
G. FENINI

L'informatica nelle amministrazioni comunali
L'amministrazione locale può trarre importanti vantaggi dall'impiego dell'informatica. L'Autore ha svolto sull'argomento una approfondita indagine statistica, di cui vengono qui riassunti i risultati.
A. BEZZOLA

Direttore Responsabile
F. Filippazzi

Comitato di Redazione
C. Falcetti, P. Lupo, N. Minnaja,
G. Occhini, M. Pardi, G. Rapelli
F. Ricci, M. Speranza

Redazione
Honeywell Information Systems Italia
Centro di Ricerca e Progettazione
20010 Pregnana Milanese (Milano)

Stampa
La Tipocromo - Milano

Autorizzazione del Tribunale di Milano
n. 93 del 20 Marzo 1974

«Quaderni di Informatica» ospita articoli e contributi di varia provenienza. Le responsabilità delle opinioni espresse rimane agli Autori.

La riproduzione di articoli della rivista, o di parte di essi, è consentita solo citando la fonte e l'autore.

PHOS: una evoluzione nelle metodologie di programmazione

MARCO MAIOCCHI
Università di Milano - Istituto di Cibernetica

ENRICO SPOLETINI
Honeywell Information Systems Italia

1 - Introduzione

La programmazione strutturata ha da tempo indicato gli strumenti linguistici e concettuali ed alcune linee guida generali da utilizzare per la costruzione di programmi di elevata qualità.
Tuttavia le indicazioni fornite da tale disciplina non permettono di costruire programmi, a partire dalle specifiche, senza che le fondamentali scelte su come organizzare il programma, e come determinare (o inventare) l'algoritmo, siano prese dal programmatore sulla base della sua esperienza e della sua sensibilità. In casi specifici, in cui gli aspetti algoritmici sono, in una certa misura, standardizzabili, sia per un approfondito studio della classe di problemi trattati, sia per la semplicità concettuale dei problemi, è possibile definire un insieme di regole operative che permettono di costruire programmi, a partire dalle specifiche, in maniera «quasi algoritmica»; si tratta cioè di seguire un certo insieme di passi che portano, naturalmente ed inequivocabilmente, alla costruzione del programma che risolve il problema assegnato, partendo esclusivamente da una opportuna ridescrizione delle sue specifiche.
In tal modo, il programmatore è in grado di dedurre la struttura del programma, ed il tipo di algoritmo, dalle specifiche del problema, anziché procedere attraverso un meccanismo di invenzione. Tale è proprio il ruolo delle metodologie di programmazione che trasformano in know-how risultati di ricerca ormai acquisiti e valutati positivamente. Nella fattispecie gli obiettivi che si riescono a conseguire sono: la velocità di costruzione dei programmi, la loro comunicabilità e la loro facile e poco costosa manutenzione, e la standardizzazione delle caratteristiche dei prodotti.
Lungi dall'essere un elemento limitativo della creatività del programmatore, una metodologia diventa strumento critico per l'esame dei problemi e delle relative analisi e contribuisce alla crescita del programmatore stesso.
Il presente lavoro espone gli obiettivi, i principi guida e gli aspetti operativi della metodologia PHOS (Program Hierarchy from Output Structure), sviluppata congiuntamente dalla Honeywell Information Systems Italia e dall'Istituto di Cibernetica dell'Università di Milano ed ora in fase di crescente diffusione.
La metodologia è stata sviluppata da un gruppo costituito, oltre che dagli scriventi, anche da T. Bettinazzi, D. Biglino, A. Maserati, F. Monzani, E. Toscani. Determinanti per la costruzione di PHOS sono state le indicazioni del prof. G. Degli Antoni. Gli autori della metodologia ringraziano C. Barassi e G. Murè per la consulenza che hanno fornito rispettivamente sulle metodologie Jackson e Warnier.

2 - Obiettivi della metodologia PHOS

La metodologia PHOS si propone di rendere possibile l'ottenimento di programmi di buona qualità, con caratteristiche standard di documentazione e codifica, mediante un procedimento definito con precisione.
Così come vedremo dettagliatamente in seguito, PHOS fornisce al programmatore strumenti concettuali per interpretare criticamente l'analisi del problema fornitagli e descrivere opportunamente alcuni elementi di essa, da cui poter dedurre la struttura del programma; vengono poi fornite indicazioni su come effettuare la codifica del programma, ancora con opportuni standards. Il risultato del procedimento è quello di ottenere programmi facilmente manutenibili, indipendentemente dal programmatore che li ha costruiti; inoltre una chiara corrispondenza tra le parti del programma e gli elementi dell'analisi che le hanno determinate rende i programmi facilmente modificabili, a fronte di cambiamenti nelle specifiche funzionali.
Tali caratteristiche dei programmi assicurano la proprietà di mantenere la propria qualità nel corso della loro vita.
Un altro effetto, di notevole portata, dell'uso della metodologia è la crescita dei programmatori che l'adottano. Gli strumenti concettuali messi a disposizione costituiscono una guida ad un esame dell'analisi (costituendo anche un'utile fase di auditing), ad uno studio critico del

modo in cui il problema deve essere descritto ed affrontato, all'individuazione di entità concettuali presenti nel problema, ma non espresse nell'analisi. L'attenzione del programmatore è completamente portata quindi su aspetti di esame del problema, di architettura generale del programma, e vengono per contro ridimensionati gli aspetti di codifica o di esame della singola sequenza d'esecuzione, che hanno invece troppa rilevanza nella programmazione tradizionale.
In particolare, l'approccio top-down seguito distribuisce scelte, decisioni e studio lungo tutto l'arco dello sviluppo del programma, garantendo l'acquisizione di una solida competenza sul problema.

3 - Caratteristiche generali della metodologia PHOS

PHOS si ispira fondamentalmente a principi di programmazione strutturata top-down. I seguenti passi logici, illustrati in fig. 1, diventano le successive fasi operative che permettono di ottenere il programma.
a. analisi e descrizione gerarchica dei dati di ingresso e di uscita: i dati in questione vengono descritti mediante raffinamenti top-down che ne dettagliano completamente la struttura sintattica in termini di sequenza, alternativa, iterazione. Tale operazione mette in evidenza la struttura logica dei dati da trattare e quindi le entità che compaiono espressamente nella descrizione come «livelli» riassuntivi di insiemi di dati più semplici, che non sempre (anzi, quasi mai) vengono espresse nell'analisi; in particolare la struttura dei dati ottenuta evidenzia le unità elementari accessibili con operazioni I/O, cioè i «records logici»;
b. deduzione della struttura del programma dalla struttura dei dati di uscita: il programma viene interpretato come «la grammatica generatrice» del «linguaggio» definito dalla «sintassi» dei risultati, ottenuta nel passo precedente, e viene quindi costruito con una struttura identica ad essa. Cioè si presuppone che la struttura generale del programma in termini di sequenze, selezioni ed iterazioni che in esso compaiono, sia legata alla struttura dei dati che esso deve produrre, in termini di successioni di dati, di alternative fra dati, e di ripetizioni di dati. In tal modo, la struttura «sintattica» dei dati di output costituisce il modello per la deduzione della struttura del programma, che ricalca completamente anche i vari livelli gerarchici della descrizione top-down.
Ogni livello di raffinamento del programma, che corrisponde ad un livello di raffinamento dei dati, viene fatto precedere e seguire da un blocco il cui scopo è di contenere istruzioni di inizializzazione e di terminazione di uno specifico trattamento;
c. verifica della coerenza tra strutture di dati d'ingresso e strutture di dati in uscita: il programma costruito secondo la struttura dei risultati può essere mantenuto con tale struttura solo se la forma con cui i dati in ingresso si presentano è coerente con quella dei risultati prodotti; questa fase ha la funzione di evidenziare eventuali difficoltà in tal senso; tali difficoltà sono chiamate «conflitti» e possono essere causati da diverse circostanze (ad es. è richiesta la stampa di informazioni in un ordine differente da quello con cui tali informazioni vengono acquisite);
d. eventuali modifiche alla struttura del programma: le situazioni conflittuali individuate al punto precedente sono classificate dalla metodologia e, per ciascun caso, è fornita una regola per modificare la struttura del programma per renderlo adeguato alla risoluzione di tali conflitti;
e. inserimento delle operazioni elementari: la metodologia fornisce indicazioni precise per l'inserimento, nei blocchi d'inizio e fine livello introdotti nella struttura del programma, di istruzioni connesse alle elaborazioni di dati: si tratta cioè di una guida alla collocazione di istruzioni di apertura e chiusura di files, di lettura, di stampa, di azzeramento e di inizializzazione di aree, di memorizzazione di valori per la gestione di iterazioni, di conteggi, ecc.; tali indicazioni hanno il significato di fornire regole standard che assicurino la coerenza delle asserzioni d'ingresso e d'uscita di ciascun modulo con quelle dei moduli di livello gerarchico superiore ed inferiore: in tal modo il programmatore, senza dover effettuare specifiche operazioni di verifica di correttezza e senza neppure conoscere il concetto di asserzione usata a scopo di verifica di correttezza, è in grado di scrivere programmi coerenti dal punto di vista delle sole operazioni volte alla gestione dei files, che, in ambienti EDP, costituisce spesso l'unica possibile difficoltà. L'allocazione di tali funzioni è suddivisa in due parti: operazioni di I/O (letture, scritture, open, close) direttamente visualizzate nella struttura, e operazioni di gestione corrente (gestione contatori, memorizzazione chiavi, ecc.).
Per le prime, le indicazioni fornite dalla metodologia sono essenziali ed estremamente precise, guidate dall'indentificazione nel programma dei punti in cui vengono trattati in input ed in output i records logici;
f. codifica del programma: il programma viene codificato mantenendone una struttura fisica gerarchica esattamente aderente a quella logica progettata, mediante un uso razionale di richiami di sottoprogrammi (ad esempio con PERFORM del Cobol). Il linguaggio utilizzato è in genere un sottoinsieme del linguaggio di programmazione a disposizione, in particolare dal punto di vista delle istruzioni di controllo; ad esempio, nel caso del Cobol, per cui sono state elaborate indicazioni non rigide ma sufficientemente precise, le sole istruzioni IF..ELSE.. e PERFORM...UNTIL... possono essere usate come strutture di controllo, e l'uso di sezioni permette l'organizzazione gerarchica. Le indicazioni sulla codifica non si limitano all'uso del linguaggio, ma anche alla commentazione: vengono fornite precise indicazioni su dove inserire i commenti e su cosa specificare in essi, al fine di ottenere una lista di compilazione il più possibile autodocumentata, cioè una lista che possa costituire la parte preponderante della documentazione dettagliata di disegno, senza la necessità di scrittura di altri documenti cartacei, con le ben note difficoltà di controllo, gestione, aggiornamento ed uso.

Gli aspetti fondamentali di PHOS sono quindi quelli di introdurre mediante una procedura di disegno l'uso di tecniche di programmazione strutturata top-down con

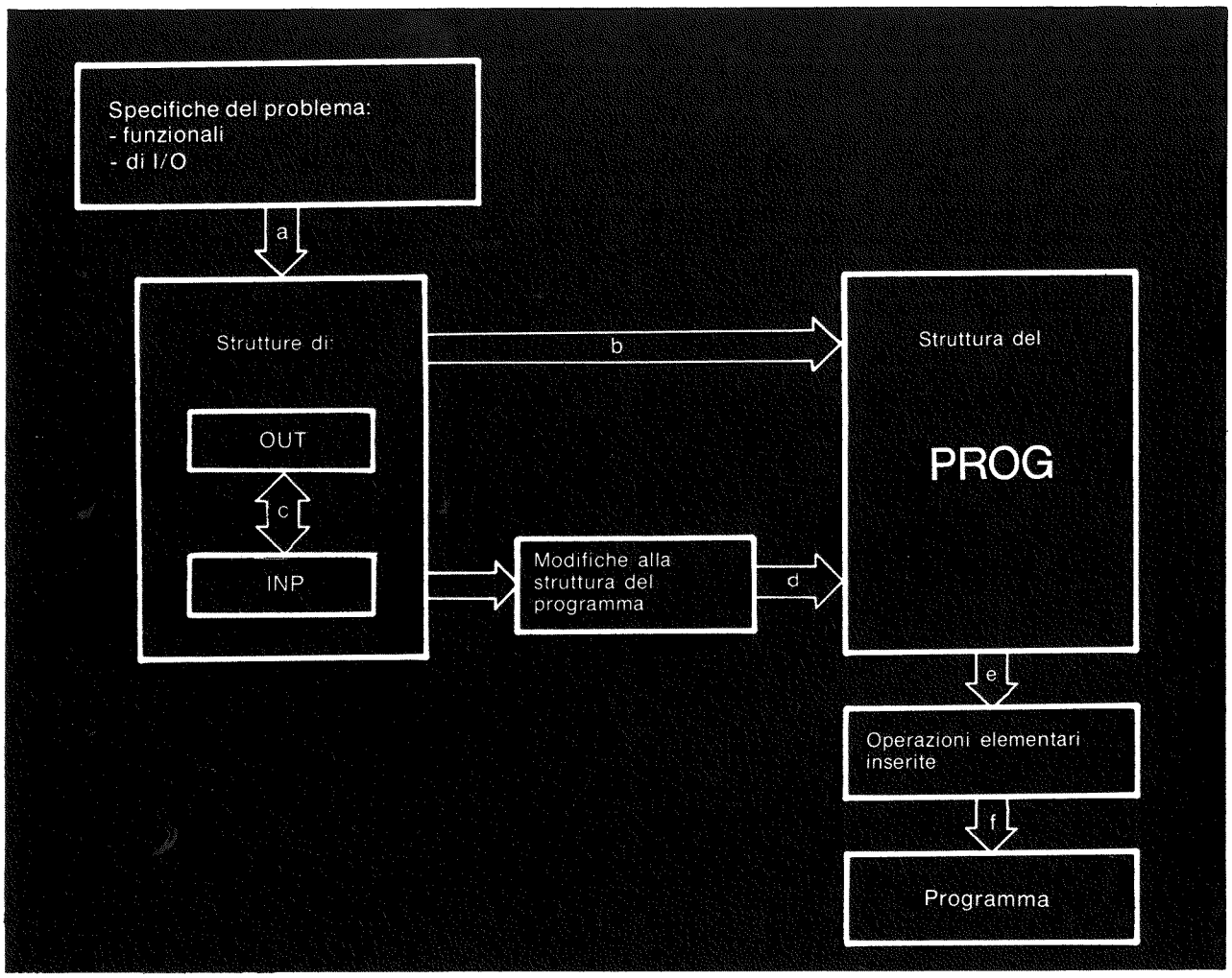


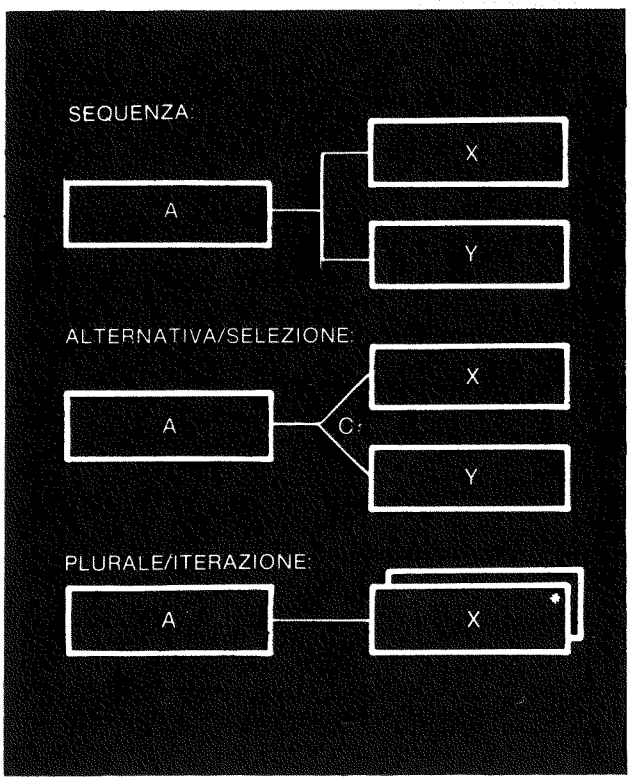
Fig. 1 - I passi della metodologia

l'implicita verifica di consistenza di asserzioni d'ingresso ed uscita dei vari moduli e con tecniche di autodocumentazione.
PHOS fornisce strumenti grafici ed indicazioni critiche su come condurre le diverse fasi.
Nel seguito analizzeremo con dettaglio, esemplificandole, ciascuna di queste fasi.

4 - Analisi e descrizione gerarchica dei dati d'ingresso ed uscita

I dati vengono descritti in una struttura in cui ogni elemento di un dato livello gerarchico può essere specificato come sequenza, alternativa o ripetizione di elementi di livello inferiore.
Allo scopo di individuare tutti gli elementi logici che intervengono nel problema, si conviene di utilizzare una sola regola di composizione per volta. La descrizione usa gli strumenti grafici rappresentati in figura 2.
Mentre diversi files d'ingresso vengono descritti separatamente, tutti i risultati, anche se fisicamente distribuiti su differenti supporti, vengono descritti in un'unica struttura, in quanto è da questa che la struttura del programma deve essere dedotta. In tal modo, la struttura del programma sarà in grado di tenere conto completamente di tutti i risultati da fornire nel loro preciso ordine.

Fig. 2 - Formalismo grafico utilizzato



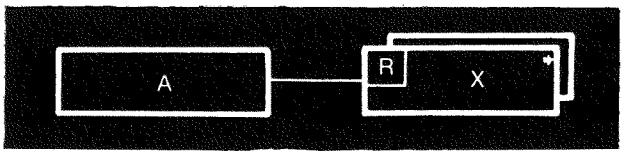


Fig. 3 - Individuazione del record logico

In questo caso la descrizione dei dati tiene conto non solo della successione fisica degli elementi da emettere, ad esempio su flussi sequenziali, ma anche dall'ordine temporale con cui i vari flussi vanno riempiti. Al contrario la descrizione dei flussi di input è separata, in quanto sarà proprio l'esecuzione del programma a determinare la corretta sequenzializzazione delle informazioni presenti su più flussi.

La descrizione dei dati riguarda tutti i flussi d'ingresso ed uscita, e quindi vengono descritti in tal modo anche i flussi ad accesso diretto; tuttavia tali flussi vengono precisati come ad accesso diretto, in quanto non avranno un ruolo preponderante nella definizione della struttura del programma, poiché le caratteristiche del loro accesso li fanno concettualmente assimilare a tabelle di memoria centrale, prive di alcuna limitazione nell'ordine d'accesso.

Nella costruzione del programma è spesso rilevante conoscere quale degli elementi logici individuati corrisponde ad una entità fisica reale, cioè ad un record: ciò viene indicato (con una R) sulla struttura dei dati come si può vedere da figura 3.

Consideriamo il seguente esempio (volutamente banale): un file di schede contenga informazioni su movimenti contabili relativi a clienti di una certa società; per ogni cliente sono presenti nell'ordine: una scheda anagrafica (TSK=1) ed eventualmente un gruppo di schede movimento (TSK=2) come dai tracciati di figura 4.

Il file sia ordinato su codice cliente e tipo scheda.

Si richiede di costruire un programma che produca la stampa di moduli (per i soli clienti con movimenti) come in figura 5.

Risultano, secondo la metodologia PHOS, le strutture di ingresso ed uscita di figura 6, in cui sono evidenziate le unità «informazioni riferite ad un cliente», iterata, e «informazioni che si riferiscono ad un singolo movimento del cliente», pure iterata.

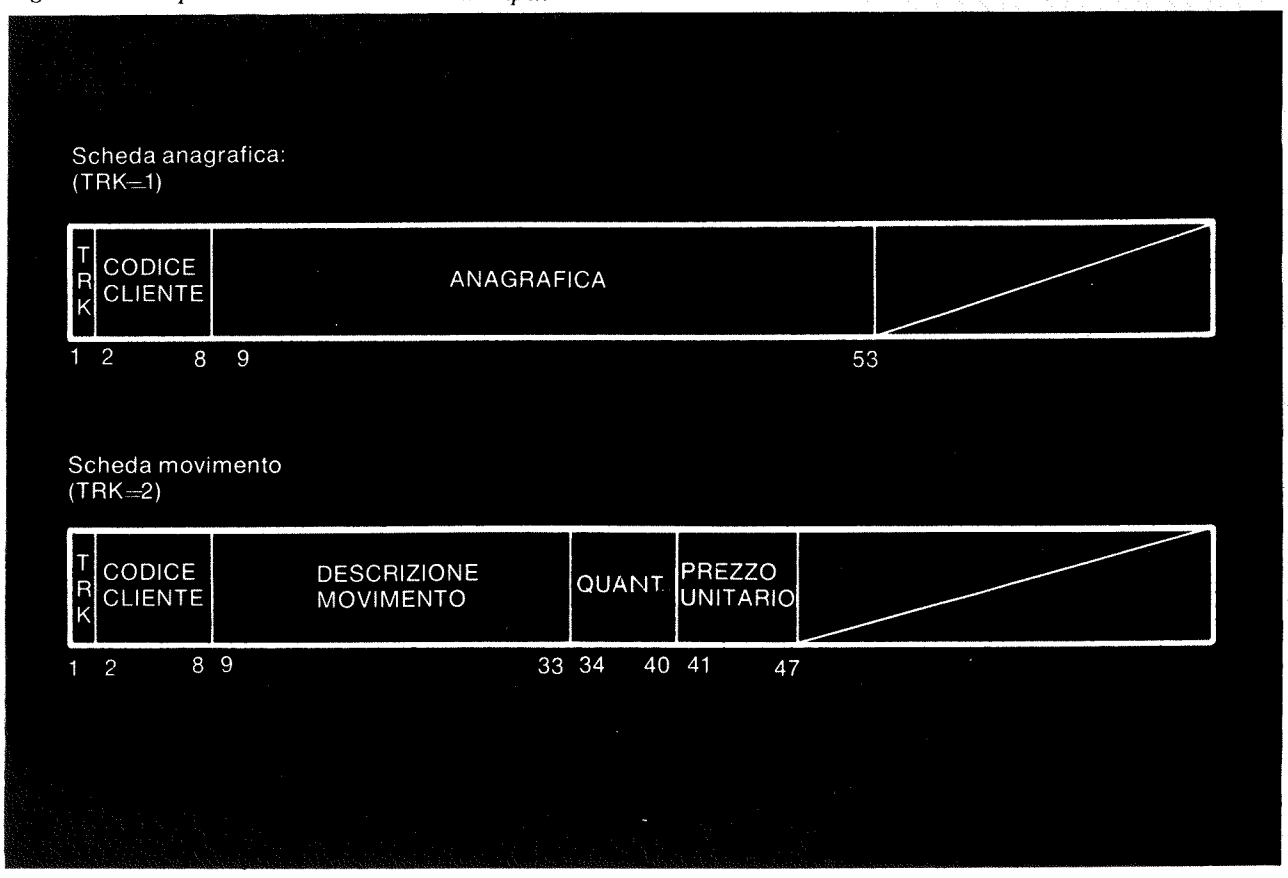
5 - Deduzione della struttura del programma

Come già indicato, la struttura del programma si ottiene, in prima approssimazione, semplicemente «ricopiando la struttura dei risultati», ed anteponendo e posponendo a quelle unità logiche raffinate al livello successivo un blocco di inizio e fine elaborazione.

Lo strumento grafico utilizzato in questa fase è lo stesso impiegato per la descrizione delle strutture di dati, ed è concettualmente identico: viene infatti qui definita, con la struttura del programma, la «struttura sintattica» delle sequenze d'esecuzione del programma stesso, cioè il loro possibile insieme.

In questa fase devono direttamente essere specificate le condizioni di selezione e di interruzione delle iterazioni, come illustrate dall'esempio 7, che fa riferimento alle strutture di dati analizzate nel precedente paragrafo.

Fig. 4 - L'esempio: tracciati dei record di input



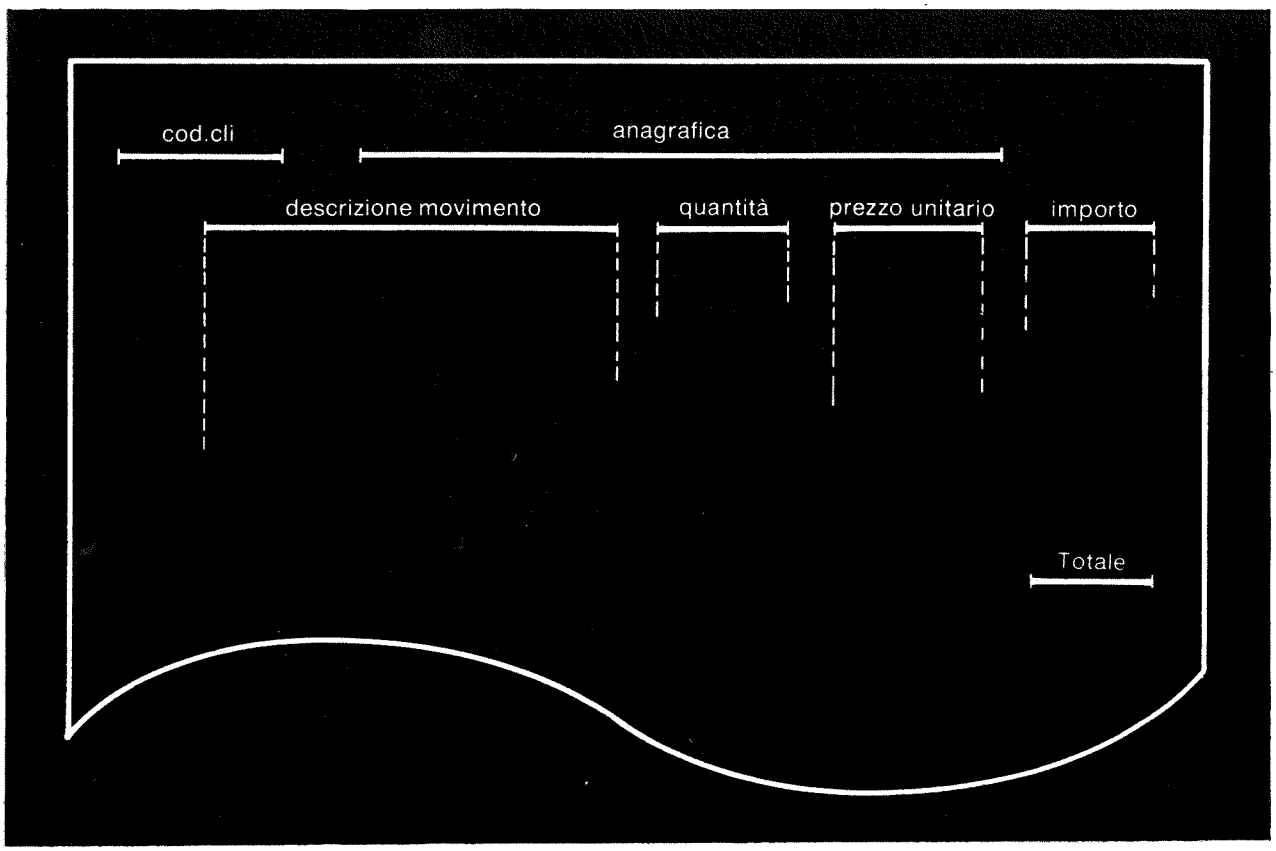
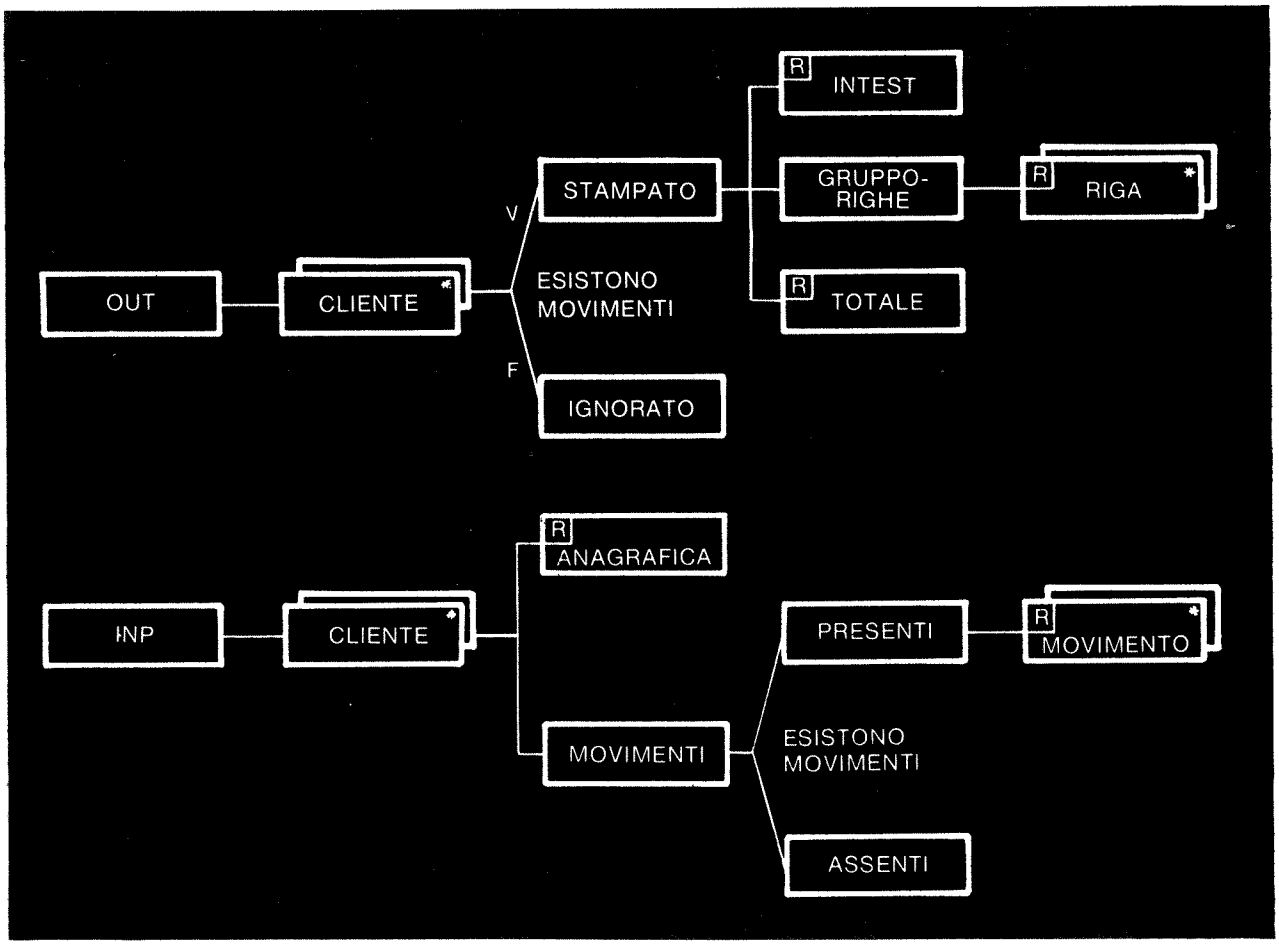


Fig. 5 - L'esempio: schema del tabulato (massimo 50 righe dettaglio per modulo)

Fig. 6 - L'esempio: struttura di input e di output



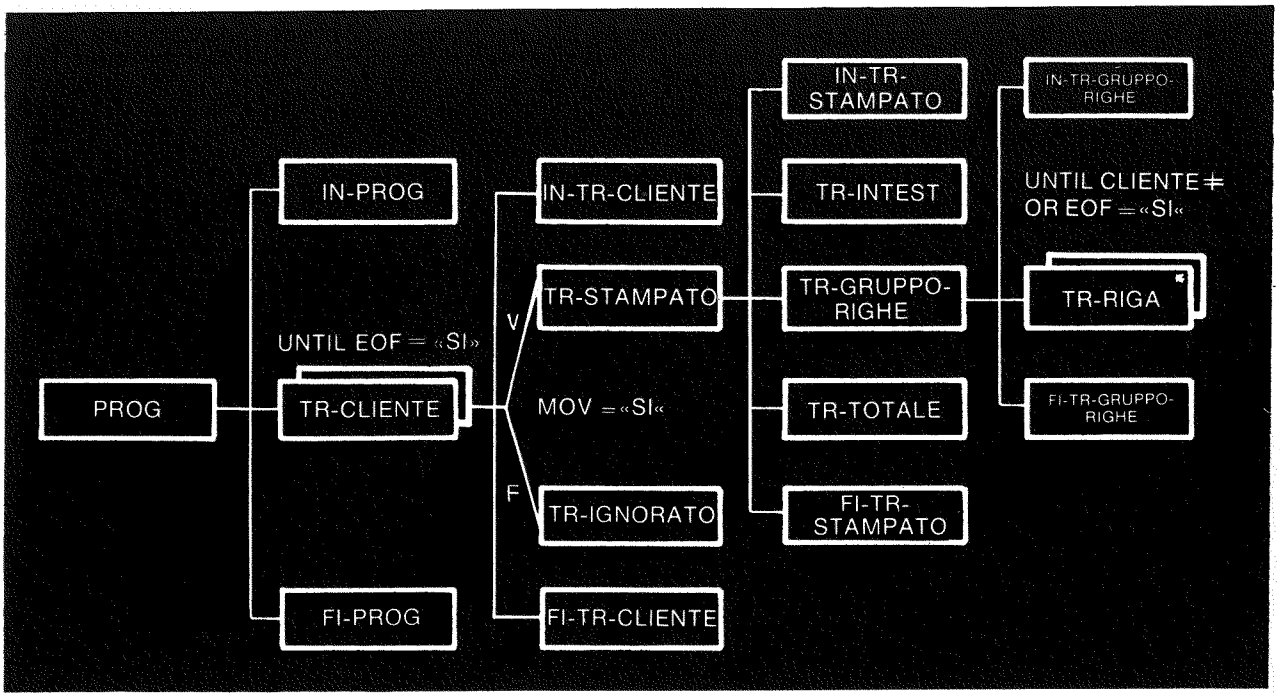


Fig. 7 - L'esempio: struttura del programma dedotta dalla struttura dell'output

6 - Analisi della consistenza tra strutture di ingresso ed uscita

L'individuazione di eventuali conflitti tra le strutture dei dati viene effettuata mediante ispezione, livello per livello, delle strutture di ingresso ed uscita.
I diversi tipi di conflitti vengono classificati come segue:

- *conflitti di indecisione,* nel caso in cui la mancata corrispondenza tra le strutture avvenga a livello di una struttura alternativa presente in output e non in input.
- *conflitti di struttura,* nel caso in cui ci sia mancata corrispondenza a livello di una struttura iterativa; in tal caso tre differenti sottocasi vengono ulteriormente individuati:
  - un'iterazione presente in input manca in output;
  - un'iterazione presente in output manca in input;
  - iterazioni presenti in input mancano in output e viceversa; spesso tali situazioni sono indicate come provenire da differente ordinamento dei dati in ingresso e in uscita, o da una differente organizzazione fisica che coinvolge considerazioni proprio sulle unità fisiche di lettura e scrittura (come elaborazione di parole con input da telescrivente che ha come unità fisica d'ingresso il carattere, ecc.).

Tale classificazione di conflitti permette che precise indicazioni vengano fornite dalla metodologia sulla base del tipo di conflitto individuato.
Nell'esempio del paragrafo 4 è facilmente individuabile un conflitto di indecisione a livello della specificazione del movimento, che in output presenta un'alternativa che non ha una corrispondenza diretta in input.

7 - Modifiche alla struttura del programma

La presenza di conflitti evidenziata dalla fase precedente indica che non è possibile, con un programma la cui struttura è derivata dai dati d'uscita, allocare le funzioni di lettura ai vari blocchi della struttura in modo da poter correttamente leggere i dati d'ingresso nell'ordine necessario. Bisogna quindi modificare il programma, e le modifiche avvengono sempre organizzando opportunamente il blocco d'inizio del livello del programma corrispondente al conflitto individuato.
Così,

- nel caso di conflitti d'indecisione, viene dettagliato il blocco d'inizio trattamento in una sequenza di un blocco che raccolga le operazioni elementari standard, e di un blocco che raccolga le informazioni necessarie per poter risolvere il conflitto, leggendo e trattando i dati d'ingresso in modo adeguato: tale blocco verrà strutturato secondo l'organizzazione dei dati d'ingresso da leggere;
- nel caso dei conflitti di struttura, in corrispondenza dei tre sottocasi precedentemente individuati, si suggeriscono tre differenti tipi di intervento:
  - nel caso di iterazioni in input assenti in output, si inserisce nel blocco iniziale del livello che presenta il conflitto una elaborazione degli oggetti in input iterati;
  - nel caso di iterazioni in output assenti in input, che presuppone un almeno parziale differente ordinamento tra ingresso ed uscita, si suggerisce di inserire nel blocco iniziale del livello di trattamento del conflitto una elaborazione che permetta il nuovo ordinamento richiesto, tipicamente la gestione di una tabella di memoria che raccolga i dati da elaborare in ordine differente, o di un file intermedio da «ordinare».
  - nel caso di iterazioni assenti sia in input che in output, si indica la scomposizione del livello di programma a cui si individua il conflitto in tre parti distinte, di cui la prima ha il compito di spezzare le unità fisiche di input in sottounità necessarie alla costruzione dell'output, la seconda ha il compito di ordinare i dati così costruiti nell'ordine opportuno, e la terza ha il compito, partendo

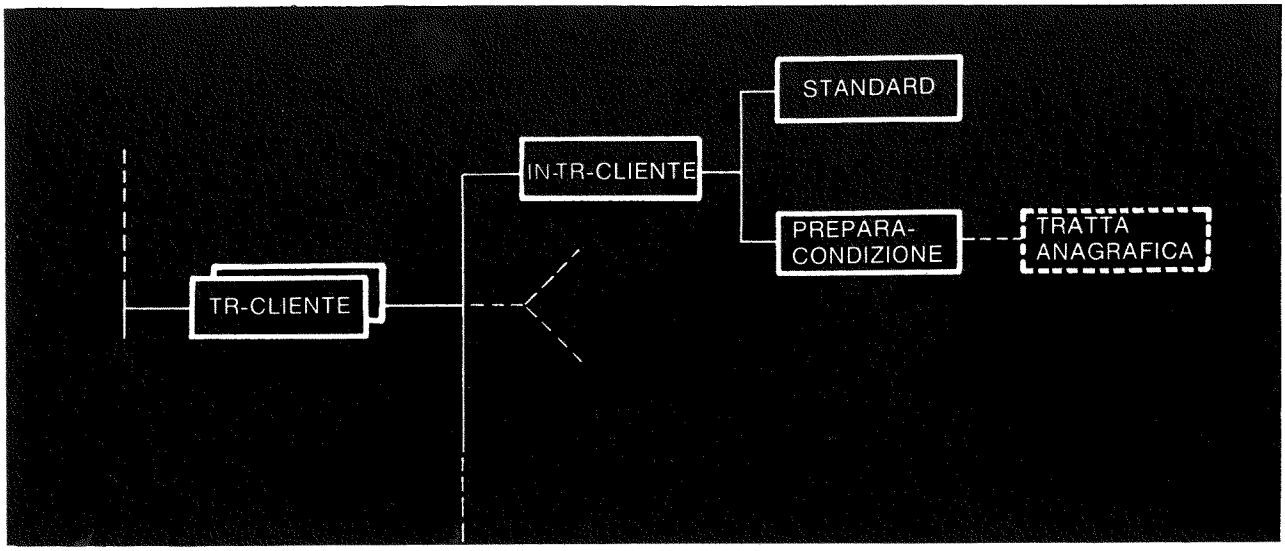


Fig. 8 - L'esempio: risoluzione del conflitto di struttura

dai dati ormai opportunamente trasformati, di produrre i risultati richiesti.

Nell'esempio dei paragrafi precedenti il conflitto di indecisione viene risolto con l'inserimento, nel blocco di inizio trattamento di un cliente, di un'elaborazione che fornisce l'informazione necessaria alla scelta seguente, e quindi di un'elaborazione che permetta di riconoscere la presenza di un movimento dopo l'anagrafica; tale elaborazione deve coinvolgere la lettura del successivo record fisico d'ingresso. La modifica operata sulla struttura è illustrata in figura 8.

8 - Inserimento delle operazioni elementari

PHOS specifica ove inserire le istruzioni di lettura, stampa, inizializzazione, ecc. lasciando al programmatore il compito di inserire nei blocchi terminali della struttura le istruzioni di calcolo vero e proprio, per effettuare il trattamento.

Ad esempio sono fornite queste indicazioni:

- nel blocco di inizio programma devono essere inserite:
  - operazioni di apertura di flussi
  - inizializzazioni di aree di lavoro, di stampa, di contatori, ecc...
  - caricamento di tabelle in memoria
  - lettura del primo record di ciascun flusso sequenziale d'ingresso
- nel blocco di fine programma devono essere inserite:
  - eventuali stampe finali,...
  - chiusure dei flussi
- nel blocco di inizio livello devono essere inserite:
  - inizializzazioni di aree locali, contatori locali, ecc.
  - ispezioni di tabelle e flussi logici random
  - memorizzazione di codici per il controllo della terminazione delle iterazioni
  - preparazione di stampe
  - ecc...
- nel blocco di fine livello devono essere inserite:
  - stampe totali locali
  - riposizionamenti di aree non locali (come «abblencamenti» di aree di stampa)
  - incrementi di contatori
  - letture successive dei flussi d'ingresso.

Ovviamente, non tutte le operazioni indicate devono essere introdotte in ciascun blocco di ogni livello: è a cura del programmatore selezionare le operazioni da inserire, usando le precedenti indicazioni come promemoria guida; è ovviamente sottinteso che, ad esempio, le memorizzazioni di codici per l'interruzione delle iterazioni vanno inserite solo nei blocchi di inizio di livelli che presentano iterazioni, le letture vanno inserite dopo che l'elaborazione dell'unità precedentemente letta è stata completata, ecc.

Vengono fornite ulteriori indicazioni di controllo, come ad esempio il controllo sul numero di letture, che deve essere, per i records di un flusso d'ingresso, n+1 (n per leggere i records effettivi e una per l'individuazione della fine flusso).

Per quanto riguarda l'esempio proposto, si osservi in figura 9 l'allocazione delle funzioni di I/O fatta direttamente sul disegno della struttura del programma e, livello per livello, l'allocazione delle funzioni elementari.

In essa si osservi la lettura aggiuntiva, introdotta, sull'indicazione della struttura dell'input, per risolvere la situazione di conflitto, che era stata precedentemente individuata.

9 - La codifica

Come già detto, la metodologia PHOS non è legata ad uno specifico linguaggio di programmazione, tuttavia essa meglio si utilizza in tutti i suoi aspetti con linguaggi di programmazione che offrano le strutture di controllo della programmazione strutturata e facilitino l'organizzazione gerarchica dei programmi.

La metodologia, per il tipo di applicazioni a cui si rivolge, è stata corredata di indicazioni sulla codifica con l'uso del linguaggio Cobol (che risulta semplice anche se questo linguaggio non è stato concepito con un preciso orientamento verso la programmazione strutturata).

Le considerazioni che seguono sono consigli per il programmatore, e non regole rigide che, se applicate senza

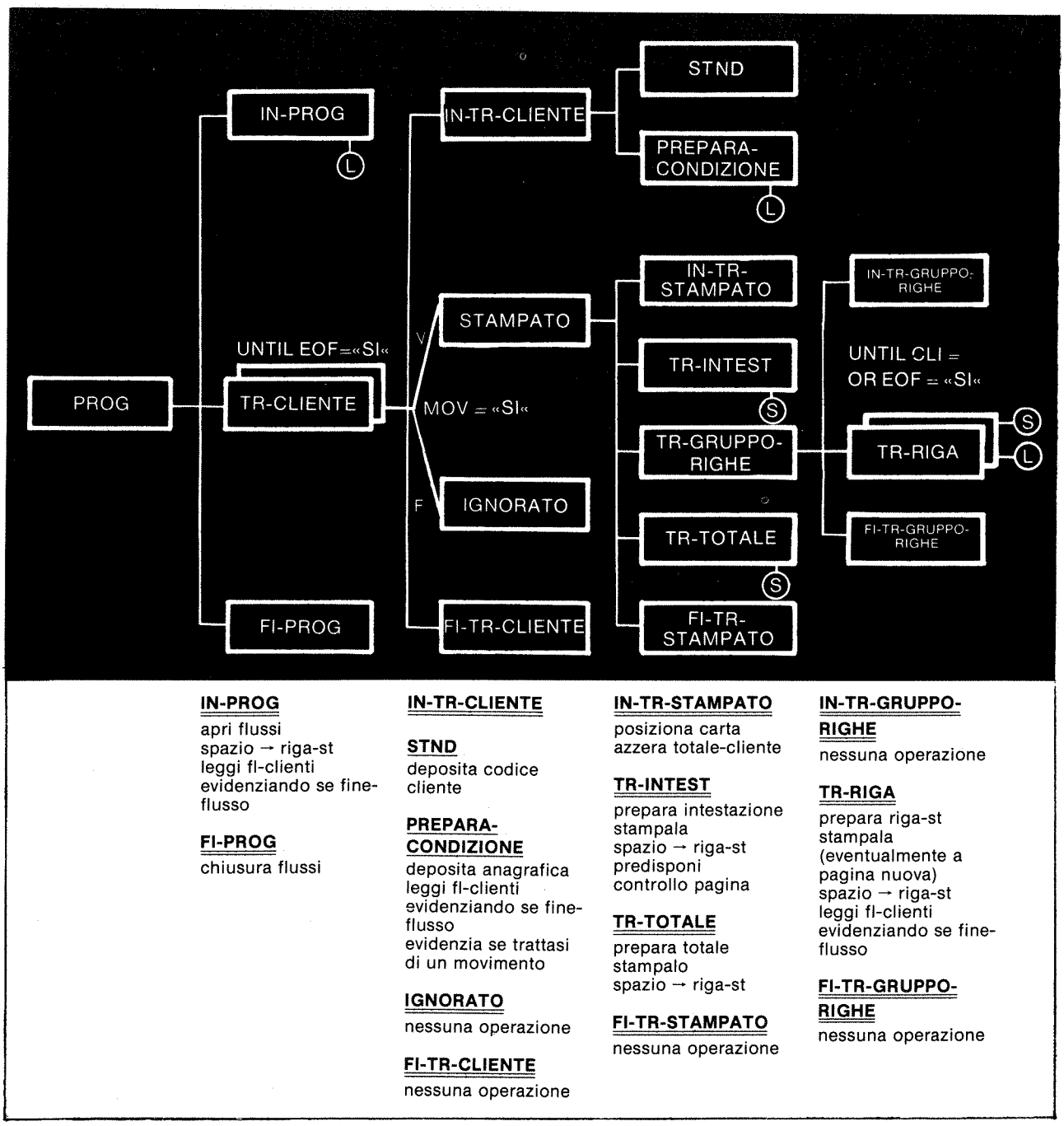


Fig. 9 - L'esempio: individuazione delle funzioni elementari

la opportuna flessibilità, potrebbero portare a programmi eccessivamente suddivisi in livelli e ridondanti nella commentazione.

L'albero che rappresenta la struttura del programma può essere considerato una SECTION Cobol (sezione principale), così come tutti i sottoprogrammi terminali da questo richiamati (tramite PERFORM); tutti gli elementi di «sostegno» nell'albero (i moduli intermedi) sono dei paragrafi nell'ambito della sezione principale: la selezione è realizzata mediante istruzioni IF ...ELSE... e l'iterazione mediante l'istruzione PERFORM ...UNTIL...

L'uso della PERFORM ...VARYING... permette di gestire agevolmente le tabelle.

Per ogni livello possono essere inseriti commenti che specificano:

- la motivazione del livello, cioè l'obiettivo che vuole raggiungere;
- le asserzioni d'ingresso e d'uscita del livello, cioè lo stato dell'elaborazione dopo l'esecuzione del livello;
- l'algoritmo impiegato, qualora questo presenti interesse o non immediatezza di comprensione;
- indicazioni varie su aree di lavoro, sulle sezioni richiamate e sul loro scopo, su note varie ritenute opportune alla comprensione del programma.

Tali indicazioni sono particolarmente significative e rilevanti nei casi di programmi di dimensioni non banali, in cui si presenti o si possa presentare l'esigenza di esaminare e sviluppare o modificare solo alcune porzioni del programma. A completamento dell'esempio, ne riportiamo in appendice la codifica in Cobol.

10 - Documentazione e manutenzione

Il procedimento consigliato da PHOS consente di evitare un pesante lavoro di costruzione a posteriori della documentazione del programma che viene preparata prima ancora di aver terminato la codifica nel linguaggio di programmazione.
Infatti tale documentazione è costituita dai disegni delle strutture dei dati e del programma e dalla codifica.
Tenendo conto del tipo di documentazione proposta, risulta estremamente semplice, ripercorrendo le linee guida del procedimento, provvedere alla manutenzione del programma e ottenere simultaneamente una documentazione aggiornata.

11 - Conclusioni

La metodologia è in corso di ampliamento, allo scopo di inquadrare in essa in modo organico altre attività inerenti alla produzione del software ed altri specifici campi di programmazione; in particolare è in corso di sviluppo un raffinamento della metodologia per una guida all'invenzione di algoritmi (secondo la visione più ampia della programmazione strutturata), una definizione delle regole procedurali per effettuare un buon testing dei programmi scritti con PHOS, un'estensione alla deduzione dei programmi in ambiente EDP in cui sia fatto estensivo uso di basi di dati integrate e/o di comunicazione di dati.
Attualmente corsi sulla metodologia vengono tenuti presso le sedi delle scuole della Honeywell che, inoltre, offre per la formazione di neo-programmatori un package in autoistruzione, dopo il cui studio è previsto un work-shop in cui vengono discussi e consolidati i concetti che gli allievi hanno appreso attraverso lo studio individuale.

Appendice

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. XYZ.
001140 ENVIRONMENT DIVISION.
001150 CONFIGURATION SECTION.
001160 SOURCE-COMPUTER. LEVEL-62.
001170 OBJECT-COMPUTER. LEVEL-62.
001180 SPECIAL-NAMES.
001190      DECIMAL-POINT IS COMMA
001200      LN5 IS IN-PAG.
001210 INPUT-OUTPUT SECTION.
001220 FILE-CONTROL.
001230      SELECT CLIENTI ASSIGN TO CLIENTI.
001240      SELECT STAMPA  ASSIGN TO STAMPA-REPORT.
002020 DATA DIVISION.
002030 FILE SECTION.
002035***** DESCRIZIONE INPUT
002040 FD   CLIENTI
002050      LABEL RECORD IS OMITTED
002060      DATA RECORDS ARE SK-ANAG SK-MOV.
002070 01   SK-ANAG.
002080      02 TSK          PIC  X.
002090      02 COD-CLI      PIC  X(7).
002100         88 EOF VALUE HIGH-VALUE.
002110      02 ANAGRAFICA   PIC  X(45).
002120      02 FILLER       PIC  X(27).
002130 01   SK-MOV.
002140      02 FILLER       PIC  X(8).
002150      02 DES-MOV      PIC  X(25).
002160      02 QUANTITA     PIC  9(7).
002170      02 PREZZO       PIC  9(7).
002180      02 FILLER       PIC  X(33).
003005***** DESCRIZIONE OUTPUT
003010 FD   STAMPA
003020      LABEL RECORD IS OMITTED
003030      DATA RECORDS ARE R-INT R-DETT.
003040 01   R-INT.
003050      02 COD-CLI-ST   PIC  BBBX(7).
003060      02 ANAGR-ST     PIC  BBBX(45)B(8).
003070 01   R-DETT.
003080      02 DES-MOV-ST   PIC  B(4)X(25).
003090      02 QUANT-ST     PIC  BBBZ.ZZZ.ZZ9.
003100      02 PREZZO-ST    PIC  BBBZ.ZZZ.ZZ9.
003110      02 IMPORTO-ST   PIC  BBBZZ.ZZZ.ZZ9.
003130 WORKING-STORAGE SECTION.
003150***** DESCRIZIONE AREE DI LAVORO
003170 77   DEP-ANAG        PIC  X(45).
003180 77   DEP-CLI         PIC  X(7).
003190 77   TOT-IMP         PIC  9(8).
003200 77   MOV             PIC  XX.
003210 77   CTR             PIC  99.
003220 77   DEP-IMP         PIC  9(8).
004006 PROCEDURE DIVISION.
004010 PROGRAMMA SECTION.
004020*****1111111*2222222*3333333*4444444*
004030 INIZIO.
004040      PERFORM IN-PROG
004050      PERFORM TR-CLIENTE UNTIL EOF
004060*                         CIOE' UNTIL COD-CLI = HIGH-VALUE
004070      PERFORM FI-PROG
004080      STOP RUN.
004100************2222222*3333333*4444444*
004110 TR-CLIENTE.
004120*             PERFORM IN-TR-CLIENTE , CIOE' :
004130                      PERFORM STND
004140                      PERFORM PREPARA-CONDIZIONE
004150             IF MOV = "SI"
004160*            PERFORM STAMPATO , CIOE' :
004170                      PERFORM IN-TR-STAMPATO
004180                      PERFORM TR-INTEST
004190*                     PERFORM TR-GRUPPO-RIGHE  , CIOE' :
004200*                             PERFORM IN-TR-GRUPPO-RIGHE
004210                              PERFORM TR-RIGA
004220                                      UNTIL COD-CLI NOT = DEP-CLI
004230*                                     CIOE' PER CAMBIAMENTO CODICE
004240*                                     CLIENTE OPP PER EOF
004250*                             PERFORM FI-TR-GRUPPO-RIGHE
004260                      PERFORM TR-TOTALE
004270*                     PERFORM FI-TR-STAMPATO
004280             ELSE NEXT SENTENCE.
004290*            PERFORM IGNORATO.
004300*            PERFORM FI-TR-CLIENTE.
```

```
004320*SOTTOPROGRAMMI SECTIONS.
004340*1111111
004350 IN-PROG SECTION.
004360 IN-P.
004370      OPEN INPUT CLIENTI
004380      OPEN OUTPUT STAMPA
004390      MOVE SPACE TO R-INT
004400      READ CLIENTI AT END MOVE HIGH-VALUE TO COD-CLI.
004410*
004420*1111111
004430 FI-PROG SECTION.
004440 FI-P.
004450           CLOSE CLIENTI STAMPA.
005020*2222222
005030*I SOTTOPROGRAMMI
005040*     IGNORATO E FI-TR-CLIENTE SONO VUOTI
005050*
005060*3333333
005070 STND SECTION.
005080 STN. MOVE COD-CLI TO DEP-CLI.
005090*
005100*3333333
005110 PREPARA-CONDIZIONE SECTION.
005120 P.  MOVE ANAGRAFICA TO DEP-ANAG
005130      MOVE "NO" TO MOV
005140      READ CLIENTI AT END MOVE HIGH-VALUE TO COD-CLI
005150                          MOVE 1 TO TSK.
005160      IF TSK = 2 MOVE "SI" TO MOV
005170      ELSE NEXT SENTENCE.
005180*
005190*3333333
005200 IN-TR-STAMPATO SECTION.
005210 IT. WRITE R-INT AFTER ADVANCING IN-PAG
005220      MOVE ZERO TO TOT-IMP.
005230*
005240*3333333
006010 TR-INTEST SECTION.
006020 TI. MOVE DEP-CLI  TO COD-CLI-ST
006030      MOVE DEP-ANAG TO ANAGR-ST
006040      WRITE R-INT BEFORE 3
006050      MOVE SPACES TO R-INT
006055      MOVE ZERO TO CTR.
006060*
006070*3333333
006080 TR-TOTALE SECTION.
006090 TT. MOVE TOT-IMP TO IMPORTO-ST
006100      WRITE R-DETT AFTER 2
006110      MOVE SPACES TO R-DETT.
006120*
006130*3333333
006140*IL SOTTOPROGRAMMA
006150*    FI-TR-STAMPATO E' VUOTO
007020*4444444
007030*IL SOTTOPROGRAMMA
007040*     IN-TR-GRUPPO-RIGHE E' VUOTO
007050*
007060*4444444
007070 TR-RIGA SECTION.
007080 TR. ADD 1 TO CTR
007090      IF CTR > 50 WRITE R-INT AFTER IN-PAG
007100                  MOVE DEP-CLI TO  COD-CLI-ST
007110                  MOVE "SEGUE" TO  ANAGR-ST
007120                  WRITE R-INT BEFORE 3
007130                  MOVE SPACES TO R-INT
007140                  MOVE 1 TO CTR.
007150      MOVE DES-MOV TO DES-MOV-ST
007160      MOVE QUANTITA TO QUANT-ST
007170      MOVE PREZZO TO PREZZO-ST
007180      MULTIPLY PREZZO BY QUANTITA GIVING IMPORTO-ST DEP-IMP
007190      ADD DEP-IMP TO TOT-IMP
007200      WRITE R-DETT
007210      MOVE SPACES TO R-DETT
007220      READ CLIENTI AT END MOVE HIGH-VALUE TO COD-CLI.
008010*
008020*4444444
008030*IL SOTTOPROGRAMMA
008040*     FI-TR-GRUPPO-RIGHE E' VUOTO
008050*
008060*FINE SOTTOPROGRAMMI
008070*
```

La comunicazione uomo-macchina e i linguaggi di programmazione

G. M. PALERMO - G. DAMAGGIO
Italsiel
Roma

1 - Introduzione

«In una elaborazione non vengono trattati oggetti del mondo reale, bensì loro rappresentazioni» (1).

Questa affermazione di Peter Wegner mostra come, nel campo dell'elaborazione automatica dei dati, un ruolo determinante sia giocato dai linguaggi e dal loro trattamento in quanto è attraverso un linguaggio che avvengono sia la descrizione degli oggetti e delle operazioni da eseguire su di essi, sia il comando e il controllo delle operazioni eseguite dal calcolatore.

Spingendo il concetto sino all'estremo, si può affermare che l'attività prevalente di un calcolatore sia quella del trattamento di linguaggi a vari livelli di rappresentazione.

Nel seguito verranno trattati due filoni di problematiche connesse a utenze diverse e alle quali corrispondono tipi di linguaggi diversi: i linguaggi programmativi e quelli di comando. Questi non hanno avuto lo stesso ritmo di sviluppo nel tempo, infatti per molti anni i calcolatori sono stati impiegati essenzialmente per scopi scientifici e utilizzati da tecnici specialisti cosicché ne è risultato privilegiato lo sviluppo teorico e pratico dei linguaggi programmativi. Solo in tempi relativamente recenti, con lo sviluppo della tecnologia nella trasmissione dei dati, si è portato il calcolatore, cioè il suo uso, a contatto dell'utente finale il quale non è tanto utente del calcolatore stesso, quanto dell'applicazione per cui questo è stato programmato.

In questa relazione verranno ulteriormente approfonditi gli aspetti riguardanti le caratteristiche dei linguaggi dai vari punti di vista: dell'utilizzo, delle proprietà intrinseche, delle tecniche di descrizione e del trattamento automatico. Inoltre per i linguaggi di programmazione verrà tracciato uno stato dell'arte dello specifico settore con una escursione nell'analisi delle possibili linee evolutive.

2 - Utenti e problemi

Prima di iniziare a parlare degli utenti dei linguaggi e dei problemi connessi con l'utilizzo di questi ultimi, cosa questa indispensabile per comprendere le forze che hanno determinato le ricerche nel campo, è opportuno fissare alcuni semplici termini:

— *linguaggio di programmazione:* è il mezzo attraverso il quale il calcolatore viene programmato cioè specializzato nello svolgimento di una specifica funzione. Si può anche dire che il linguaggio di programmazione serva per trasformare il calcolatore da risorsa generica a risorsa specifica,

— *linguaggio di comando:* è il mezzo attraverso il quale il calcolatore, preventivamente programmato per una specifica funzione, viene comandato alla esecuzione della funzione stessa e controllato secondo le modalità operative previste (fig. 1).

Avendo individuato queste due grandi categorie di linguaggi di comunicazione uomo-macchina possiamo ora analizzarne le caratteristiche specifiche in termini di: classi di utenza, classi di linguaggio e loro caratteristiche.

È opportuno per il prosieguo dell'analisi, partire dall'esame dell'utenza di questi linguaggi in quanto questa, per sue caratteristiche intrinseche ed esigenze specifiche, ha in larga misura determinato l'attuale sviluppo dei linguaggi per ciascuna delle due categorie individuate: in particolare l'utente programmatore per i linguaggi di programmazione e l'utente non-programmatore per i linguaggi di comando.

L'utente programmatore possiede in genere le seguenti caratteristiche:

— è un tecnico di elaborazione automatica dei dati;

— non è un esperto di una singola applicazione ma è in grado di produrre programmi per un insieme vario di applicazioni;

— ha come fine la produzione di programmi e il loro mantenimento nel tempo in quanto "beni durevoli" di alto costo;

— lavora di solito in gruppi di lavoro nei quali il prodotto finale (il programma) viene ottenuto attraverso l'assiemaggio di parti sviluppate separatamente dai vari

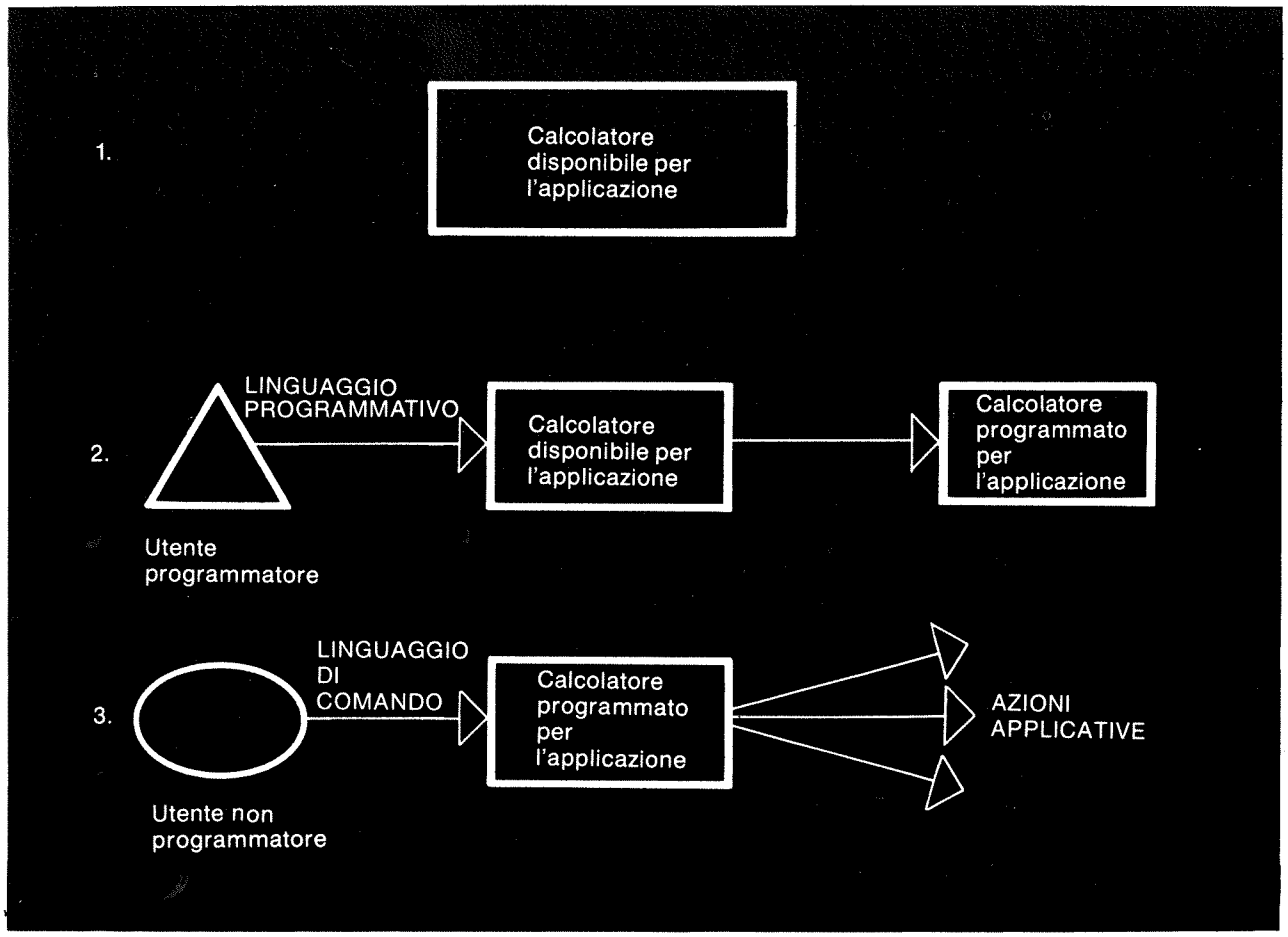


Fig. 1 - Linguaggio di programmazione e linguaggio di comando

componenti il gruppo di lavoro;
— tende alla ottimizzazione del proprio prodotto in termini di risorse impiegate (tempo di elaborazione, occupazione di memoria centrale e periferica, impegno di linee di trasmissione, etc.).
Ciascuna delle suddette caratteristiche ha avuto una parte di rilievo nello sviluppo dei linguaggi di programmazione e dei relativi mezzi di trattamento ma ne ha costituito nel contempo anche il limite di sviluppo.
La programmazione dei primi calcolatori avveniva in "linguaggio macchina" utilizzando direttamente il codice interno del calcolatore per specificare le sequenze di istruzioni desiderate; da allora, con l'obiettivo di rendere la programmazione accessibile ad un numero sempre crescente di addetti minimizzando i costi di addestramento e aumentandone la produttività, si è assistito allo sviluppo di linguaggi di sempre più facile uso.
Tali linguaggi, detti ad "alto livello", non richiedono infatti una profonda conoscenza della struttura interna di un calcolatore e dei meccanismi che ne regolano la variazione nel tempo degli stati, permettono la riduzione dei tempi di scrittura e di messa a punto dei programmi, aumentano la "leggibilità" dei programmi stessi rendendo possibile l'intervento su di essi da parte di persone diverse e ne facilitano la "trasportabilità" da un calcolatore ad un altro.
Il raggiungimento dei vantaggi sopraesposti ha fatto insorgere la speranza di poter innalzare il livello di tali linguaggi sino a quello del linguaggio naturale corrente; questa speranza è stata da tempo accantonata in quanto l'esigenza di sganciarsi dalla programmazione in linguaggio macchina verso linguaggi di più alto livello, si accompagna altresì alla condizione irrinunciabile di poter mantenere un rigoroso e non ambiguo mezzo di espressione delle funzioni che il calcolatore dovrà espletare.
Gli attuali linguaggi di programmazione, in particolare i linguaggi procedurali, quelli cioè che descrivono le sequenze logiche in cui si articolano gli algoritmi di soluzione di un problema, sono pertanto un opportuno compromesso tra opposte esigenze ed hanno oramai raggiunto un grado di relativa stabilità sia nella struttura sintattica che nella potenzialità di espressione semantica; crediamo quindi di poter escludere che ulteriori sviluppi ed evoluzioni di questi linguaggi possano avvenire attraverso una mera "liberalizzazione" delle strutture sintattiche.
L'utente non-programmatore ha per contro le seguenti caratteristiche:
— non è un tecnico di elaborazione automatica dei dati,
— è un esperto in una o più specifiche applicazioni,
— ha necessità di estrarre ed aggiornare in modo semplice e rapido i dati da una base informativa di cui conosce il contenuto ma non la strutturazione interna al calcolatore;
— ha necessità di attivare e dirigere una o più elabora-

zioni in funzione di eventi esterni attinenti alla specifica applicazione.
Per questo tipo di utente si sono definiti linguaggi che mantengono il più stretto legame con le caratteristiche del linguaggio corrente, sia verbale che gestuale (*), che l'utilizzatore impiega nell'espletamento delle proprie attività. Peraltro, data l'elevata specificità applicativa che comporta una alta ripetitività di strutture sintattiche semplici (gergo applicativo), tali linguaggi sono in ultima analisi un ristretto sottoinsieme di quello naturale e sono decisamente non procedurali.
Essi sono tuttora in forte evoluzione in funzione della crescente diffusione nell'uso dei calcolatori presso utenti non specialisti e della progressiva e continua riduzione del costo di elaborazione; questo ha fatto si che possano essere soddisfatte le esigenze di una tale utenza predisponendo modi di comunicazione sempre più semplici e sempre più "naturali" per l'utente stesso anche se ciò comporta un elevato impegno di risorse elaborative.
La scelta del linguaggio naturale, che come accennato non si è rivelata congeniale all'utilizzo da parte del programmatore, è in questo ambito praticabile in quanto viene incontro alle esigenze di un utente non esperto in elaborazione automatica dei dati. Si tratta in generale di linguaggi orientati ad uno specifico campo di applicazione e pertanto risulta superabile, seppure con un considerevole impegno di risorse elaborative, il problema delle ambiguità tipiche del linguaggio naturale; ciò è possibile sia per la specificità del linguaggio usato — gergo applicativo — sia per l'impiego di modalità di colloquio interattive tra l'utente e la macchina, la quale può essere predisposta per "guidare" ed "assistere" l'interlocutore utente nella formulazione corretta dei comandi.

3 - Sviluppo dei linguaggi di programmazione

Il primo linguaggio programmativo fu sicuramente il "linguaggio macchina" del primo calcolatore a programma memorizzato che, come ogni calcolatore elettronico basato sulla macchina di Von Neuman, era capace di interpretare, cioè di eseguire, solamente "istruzioni macchina" (3).
L'insieme di tali istruzioni viene solitamente scelto in sede di progettazione in base a: considerazioni relative allo spazio di memoria occupato dai futuri programmi, facilità di interpretazione da parte di circuiti che siano relativamente poco complessi e quindi veloci, ecc.; quindi problemi di prestazioni fanno sì che, nella definizione del linguaggio macchina, pochissimo ci si preoccupi della sua usabilità da parte di esseri umani: ne risulta la estrema difficoltà nell'impiegarlo per la scrittura di programmi complessi.
Si utilizza piuttosto il calcolatore stesso ed appositi programmi (i compilatori) per tradurre in linguaggio macchina (detto anche linguaggio oggetto) programmi scritti in altri linguaggi (detti linguaggi sorgente); è appunto a questi ultimi che ci si riferisce in genere con l'appellativo di linguaggi programmativi.
I più semplici linguaggi studiati per essere utilizzati direttamente dall'uomo sono quelli assemblativi (4). Essi sollevano il programmatore dal problema della determinazione della posizione esatta in memoria dei dati e delle istruzioni e della loro codifica interna, permettendo di indicarne valore e posizione in termini simbolici. Detti anche "orientati alla macchina", tali linguaggi sono tutt'oggi utilizzati per applicazioni ove si richiedano particolari prestazioni (programmi che debbono essere eseguiti moltissime volte o che debbono sfruttare peculiarità della macchina).
I linguaggi assemblativi costituiscono uno dei primi strumenti di programmazione, essi però non rappresentano il vero punto di svolta per quel che concerne la programmazione dei calcolatori elettronici, lasciando all'uomo la definizione di moltissimi dettagli della macchina durante il suo tentativo di risolvere un certo problema.
Il primo linguaggio, accolto con indifferenza e scetticismo, che realmente costituisce una pietra miliare nel campo dei linguaggi di programmazione fu il FORTRAN (5). Esso fu talmente rivoluzionario che alla sua comparsa — 1958 — si parlò di programmazione automatica; di fatto il FORTRAN, in una sua più recente versione, è uno standard per la programmazione scientifica e tecnica di grandi e piccoli calcolatori: programmi scritti in FORTRAN, con particolari cure, sono trasportabili sulla maggioranza delle macchine esistenti con poche o nessuna modifica.
Poco dopo il FORTRAN nacque l'Algol 60 (6), la cui importanza teorica fu sicuramente notevole; moltissimi concetti contenuti nell'Algol 60 li ritroviamo in linguaggi oggi sotto studio sia negli Stati Uniti che in Europa e destinati a divenire i futuri linguaggi universali per gli anni '80. Tutt'oggi derivati dall'Algol o suoi "dialetti" sono comunemente impiegati in testi di scienza dei calcolatori per la descrizione di esempi e di algoritmi.
Ancora in questo periodo, primi degli anni '60, fu progettato il linguaggio programmativo più usato nel mondo e nel quale è scritta la maggior parte dei programmi applicativi esistenti: il COBOL (7).
Oltre al merito di aver posto in luce il problema della definizione dei dati da elaborare come almeno paritetico a quello delle procedure da eseguire, il COBOL ebbe il grosso pregio di essere scelto dal Dipartimento della Difesa degli USA, il maggior consumatore mondiale di elaborazione automatica dei dati, come linguaggio obbligatorio per i suoi calcolatori. Quanto detto, oltre ad una notevole comprensibilità anche da parte di persone poco addestrate, fece di fatto del COBOL uno standard mondiale per la programmazione applicativa e gestionale (8).
Si può ben dire che il FORTRAN, l'Algol 60 ed il COBOL costituiscano la parte più significativa, sia concettualmente che in termini di diffusione, dei linguaggi ad alto livello della prima generazione; essi sono tutt'oggi come si è detto i più diffusi (9).
Per comprendere meglio le problematiche connesse alla seconda generazione di linguaggi, quella degli anni '70, occorre soffermarsi su quanto segue.

(*) Questi linguaggi hanno come elementi fondamentali non solamente caratteri di un alfabeto, ma anche posizioni di oggetti fisici nello spazio; la comunicazione con la macchina avviene sia attraverso una comune tastiera alfanumerica, sia attraverso particolari dispositivi quali light-pen, digitizer, joy-stick, etc. (2)

Un aspetto importantissimo per la diffusione di un linguaggio programmativo è senz'altro la sua "stabilità", cioè la probabilità che esso resti diffuso ed assistivo dai costruttori per molti anni. In fatti gran parte dell'investimento in elaborazione automatica dei dati risiede nei programmi, i quali praticamente non sopravvivono alla morte del linguaggio nel quale sono scritti. D'altronde, opposta a questa forza conservatrice, nasce l'esigenza di sopperire alle limitazioni dei linguaggi della prima generazione, limitazioni che si sono andate via via delineando man mano che venivano fatti progressi nella scienza dei calcolatori. Ciò è assieme la causa della nascita e della scarsa diffusione dei linguaggi della seconda generazione.

Il PL/1 (10) nato con l'intento di realizzare un linguaggio "universale" adeguato ad applicazioni sia scientifiche, sia commerciali, sia di sistema, è stato ottenuto fondendo insieme le caratteristiche giudicate migliori del FORTRAN, del COBOL e dell'Algol 60. Ne è risultato un linguaggio molto complesso e privo di una sua integrità concettuale. Gli indubbi vantaggi rispetto ai linguaggi precedenti ed il considerevole appoggio da parte della IBM non sono stati peraltro sufficienti a diffonderlo ampiamente; in tutto ciò una parte non indifferente è stata infine svolta dalla freddezza con cui gli altri costruttori ed il Dipartimento della Difesa USA lo hanno accolto.

Il secondo filone perseguito nel superamento dei problemi tipici dei linguaggi della prima generazione è rappresentato dall'Algol 68 (11), ottenuto mediante sistematica generalizzazione dei costrutti dell'Algol 60; il risultato di un tale sforzo non è però stato premiato da una corrispondente diffusione. Ciò è dovuto a svariati motivi di natura diversa: da una parte il linguaggio è molto potente ma troppo generale e flessibile per essere compreso appieno dal programmatore applicativo (ad esempio una possibilità dell'Algol 68 è la definizione di nuovi costrutti sintattici sia per i dati che per gli operatori), dall'altra non esiste una documentazione adeguata e soprattutto non ne sono stati realizzati compilatori per un sufficiente numero di macchine. Infine, a differenza del PL/1 che può essere appreso solo per le parti di interesse, l'Algol 68, per la sua integrità concettuale, non si presta a tale modalità di apprendimento.

Il terzo importante filone che ha condotto a linguaggi della seconda generazione è stato quello basato sulla applicazione esplicita del concetto di "astrazione" (12).

Con ciò si intende la caratterizzazione di un oggetto mediante un insieme di attributi che ne descrivono il comportamento, astraendo quindi la parte essenziale dai dettagli realizzativi. Un linguaggio basato su tale meccanismo permette e suggerisce la costruzione di oggetti dei quali siano ben distinte le caratteristiche "esterne" o pubbliche (note ad altri oggetti che lo utilizzano) da quelle "interne", o private, relative a dettagli realizzativi dell'oggetto stesso.

Il linguaggio più importante di questo terzo filone è senz'altro il PASCAL, descritto da N. Wirth (13); altri, basati su medesimi concetti ma sviluppati allo scopo di mettere in luce caretteristiche particolari, sono il CLU (14) e l'ALPHARD (15).

Il punto di maggior forza dei linguaggi della famiglia del PASCAL è sicuramente il meccanismo dell'astrazione che è stato un passo avanti nell'importante compromesso semplicità/potenza. Come vedremo in seguito, infatti, il problema cruciale degli anni '80 sarà probabilmente quello della produzione di programmi a bassi costi, quindi della pragmatica dei linguaggi programmativi; una ragione importantissima per cui si prevede che i linguaggi tipo PASCAL saranno la base per la terza generazione è proprio la loro migliore "usabilità".

Tra le caratteristiche che implicano questa usabilità, citiamo:

— verifica del corretto accesso ai dati da parte del compilatore stesso;

— possibilità di verifiche di correttezza dei programmi; (16).

— orientamento alla programmazione strutturata (17).

Accanto a questa linea evolutiva, sono sorti altri linguaggi o classi di linguaggi con caratteristiche varie e volte a soddisfare diverse esigenze.

Tra i linguaggi per applicazioni non numeriche, ed in particolare relative alla intelligenza artificiale, il LISP (18) riveste un ruolo particolare sia per la sua semplicità che per la sua potenza.

Fondato sul calcolo il LISP è tutt'oggi utilizzato come base ed esempio di applicazioni di teorie matematiche della computazione, in particolare della semantica dei linguaggi programmativi.

Ancora in questa classe di linguaggi si possono citare lo SNOBOL (19) e il COMIT (20) ambedue per il trattamento del linguaggio naturale.

Una seconda importante classe, orientata questa volta a specifiche applicazioni, è costituita dai linguaggi non-procedurali ad altissimo livello per particolari problemi (es.: NASTRAN, SEPOL, NOMAD). Questi linguaggi constano di istruzioni per la descrizione di un sistema (in genere in una classe ristretta di sistemi; ad esempio, nel caso dell'ingegneria civile, strutture reticolari composte di aste e nodi) e di poche potenti istruzioni per la produzione di alcuni risultati (sforzi in vari punti della struttura, etc.), non si parla ovviamente di programmazione automatica poiché la potenza e l'enfasi sull'aspetto descrittivo del linguaggio sono state pagate a prezzo di una drastica riduzione di generalità.

Di linguaggi di questo tipo ne esistono moltissimi per svariate applicazioni i più complessi dei quali sono veri e propri linguaggi ed i meno complessi sono solamente i comandi di programmi applicativi molto generali.

Infine non si può non accennare all'APL (A Programming Language) di Iverson (21) che è diventato molto popolare tra gli utenti di sistemi interattivi per applicazioni tecniche; dotato di istruzioni molto potenti, che tra l'altro permettono di operare con facilità su vettori e matrici, e studiato particolarmente per essere utilizzato in ambiente interattivo, l'APL non sembra essere il futuro per i grossi sistemi, ma è piuttosto un'ottima soluzione per l'utente esperto di calcolatori, il ricercatore, il tecnico, per chi necessita di un rapido e facile accesso alla macchina per risolvere problemi specifici.

GENERATORE DI LINGUAGGIO	e	RICONOSCITORE DI LINGUAGGIO
1 - GRAMMATICA DI TIPO O (unrestricted)		MACCHINA DI TURING
2 - GRAMMATICA DI TIPO 1 (context-sensitive)		AUTOMA «linear bounded»
3 - GRAMMATICA DI TIPO 2 (context-free)		AUTOMA «push-down»
4 - GRAMMATICA DI TIPO 3 (finite-state)		AUTOMA FINITO

Fig. 2 - Relazioni di Chomsky tra grammatiche ed automi

4 - Descrizione e trattamento automatico dei linguaggi

Esaminiamo ora alcuni aspetti e concetti riguardanti i linguaggi in generale.

La teoria dei linguaggi, naturali e non, si è sviluppata negli anni '50 e '60 sulla base dei lavori del linguista Noam Chomsky il quale ha affrontato il problema definendo un linguaggio, basato su un dato alfabeto, come l'insieme delle frasi valide formate componendo i simboli dell'alfabeto (simboli base) e studiando le regole necessarie per riconoscere se una frase appartenga o meno ad un determinato linguaggio (22).

In particolare Chomsky ha definito un insieme di relazioni tra una gerarchia di "grammatiche" – meccanismi di generazione delle frasi del linguaggio – e una gerarchia di "automi" – meccanismi di riconoscimento delle frasi del linguaggio (v. fig. 2). Tra queste, le relazioni, riguardanti grammatiche e automi di tipo 0 e 1, cioè a più alto livello gerarchico, sono maggiormente legate al linguaggio naturale e di portata essenzialmente teorica, mentre le relazioni tra grammatiche e automi di tipo 2 e 3 a più basso livello gerarchico, hanno avuto un riflesso pratico nelle ricerche sui linguaggi di programmazione ad alto livello e nello sviluppo dei relativi compilatori (23).

Particolare rilevanza ha avuto a questo proposito nello sviluppo della teoria dei linguaggi di programmazione l'Algol 60 la cui definizione generativa da una grammatica "context-free" ha poi condotto da un lato alla specificazione generativa della sintassi dei successivi linguaggi di programmazione e dall'altro lato l'impiego sistematico nello sviluppo dei compilatori di automi "push-down" (o stack).

Nel descrivere un linguaggio occorre distinguere tra due classi di proprietà: le proprietà sintattiche e le proprietà semantiche, riconducibili rispettivamente ai problemi di descrizione e di definizione di un linguaggio.

Il primo problema viene risolto mediante regole che definiscono quali sono i simboli base del linguaggio e quali sono le corrette sequenze di tali simboli che ne fanno parte; questo ha comportato la ricerca di una notazione formale in cui esprimere la sintassi di un linguaggio.

Una tale notazione detta metalinguaggio, è stata introdotta da Backus nel '59 nella descrizione dell'Algol 60 (Backus Normal Form) ed è essa stessa riconducibile, seppure con una diversa notazione, ad una delle grammatiche di Chomsky. Nella fig. 3 è riportato un semplice esempio di impiego del metalinguaggio di Backus che ha il seguente significato:

– la formula metalinguistica a) dice che una <parte sinistra> ha la forma di una <variabile> seguita dal simbolo:=

– la formula metalinguistica b) dice che una <lista di parte sinistra> ha la forma di una <parte sinistra> op-

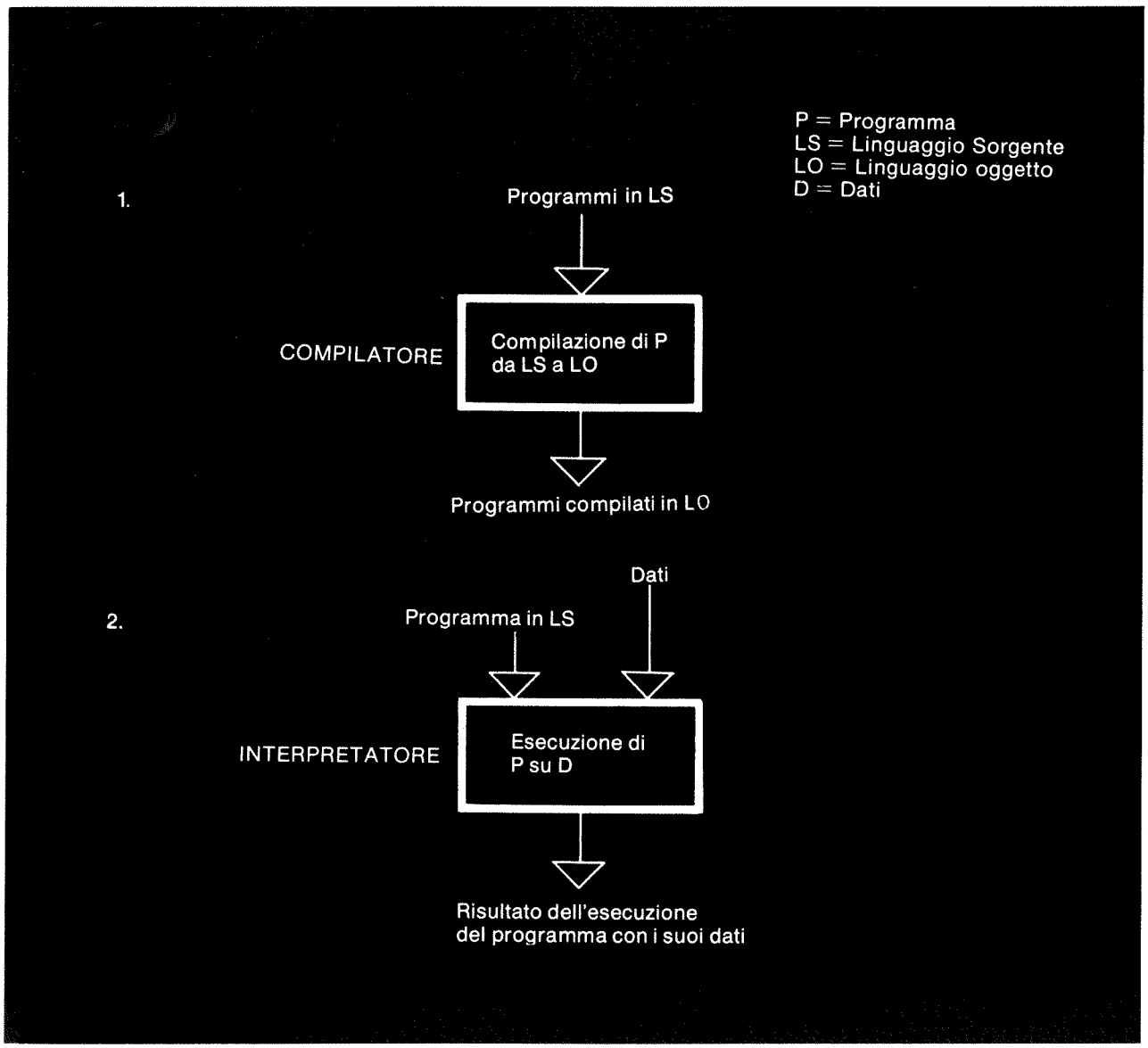
Fig. 3 - Esempio di sintassi della frase di assegnazione dell'Algol 60 in BNF

a) < parte sinistra >	::= < variabile > : =
b) < lista di parte sinistra >	::= < parte sinistra > **I** < lista di parte sinistra > < parte sinistra >
c) < frase di assegnazione >	::= < lista di parte sinistra > < espressione aritmetica > **I** < lista di parte sinistra > < espressione booleana >

pure di una <parte sinistra> seguita da una <parte sinistra> ripetuta quante volte si vuole,
– la formula metalinguistica c) dice che una <frase di assegnazione> ha la forma di una <lista di parte sinistra> seguita da una <espressione aritmetica> oppure da una <lista di parte sinistra> seguita da una <espressione booleana>.
Pertanto le seguenti sono tutte corrette frasi di assegnazione:
n:= n+1
s:= p[o]:= n:= n+1+s
A:= B/C -V-9xS
S[v, k+2]:= 3-arctan (s x zeta)
V:= Q>Y ∧ Z.
Il problema della definizione di un linguaggio consiste nell'associare ad ogni sequenza di simboli sintatticamente corretta un "significato" che, come vedremo meglio in seguito, può essere espresso associando ad un costrutto del linguaggio sorgente o un costrutto del linguaggio oggetto ovvero l'azione di una "macchina".
La definizione formale della semantica del linguaggio è l'argomento che maggiormente ha impegnato i ricercatori negli ultimi anni e rappresenta molto di più che il passo logico successivo rispetto alla descrizione formale della sintassi (24), (25), (26).

Occorre a questo punto premettere al prosieguo della relazione che proprio sull'aspetto semantico sorgono anche le maggiori difficoltà al trattamento automatico del linguaggio naturale, pur se negli ultimi anni, dopo la delusione seguita ai facili entusiasmi degli anni '50 e '60, si sono intensificate, e con risultati di crescente interesse, le ricerche nel campo; al contrario, operando i linguaggi di programmazione in un universo di significati estremamente più ridotto rispetto al linguaggio naturale, i tentativi di affrontare il problema della definizione di tali significati sono stati più numerosi e sul piano teorico si possono considerare riusciti (27).
Gran parte dei concetti base sono stati messi a fuoco negli anni '60 e '70 da John McCarthy con la funzione APPLY del linguaggio LISP e dal laboratorio IBM di Vienna con il "Vienna Definition Language" (28).

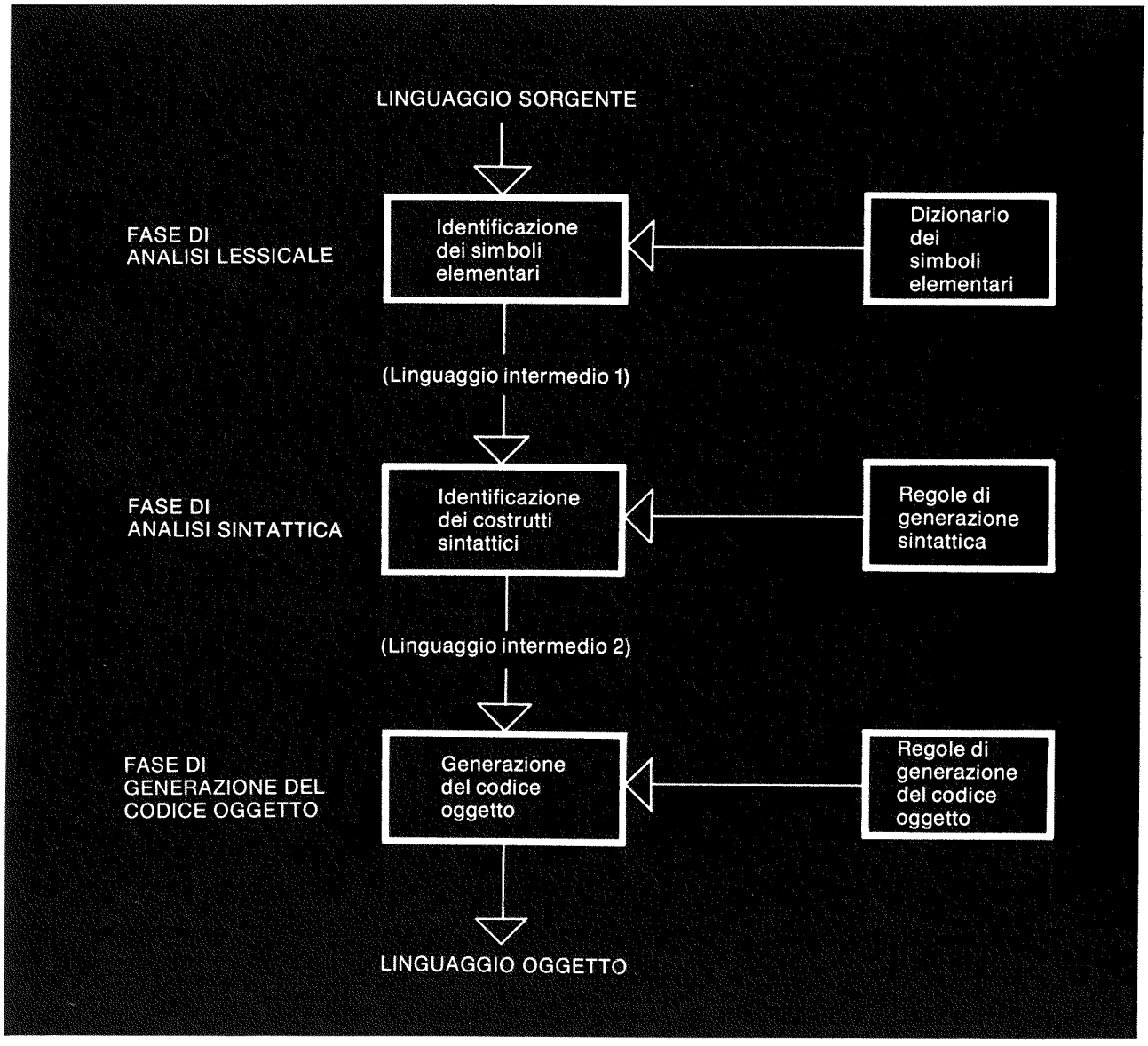
Fig. 4 - Schemi di definizione semantica di un linguaggio di programmazione



Il problema della definizione semantica di un linguaggio di programmazione ad alto livello può essere affrontato come già accennato secondo due diverse modalità:
– un linguaggio può essere definito mediante un "interpretatore" cioè un meccanismo dinamico (un programma) che definisce il significato di un programma in linguaggio sorgente in termini di azioni di una macchina su un determinato insieme di dati;
– un linguaggio può essere definito mediante un "compilatore", cioè un meccanismo di trasformazione statico che definisce il significato di un programma in linguaggio sorgente in termini di sequenze di istruzioni compilate in un dato linguaggio oggetto di cui sia nota la semantica (fig. 4).
Sia la funzione APPLY del LISP che il "Vienna Definition Language" sono esempi di definizione di un linguaggio mediante interpretatore; in entrambi i casi un insieme di programmi definisce in modo rigoroso la semantica di ogni costrutto sintattico in linguaggio sorgente.

L'Algol 60 era definito in termini di interpretatore anche se espresso in termini non formali ma mediante l'uso di espressioni verbali in lingua inglese corrente, mentre EULER, un dialetto dell'Algol 60, è un esempio di definizione semantica mediante compilatore (29).
Nel caso del LISP esso era definito mediante la funzione LISP APPLY, la quale altro non è se non un programma LISP che, avuto in ingresso un programma scritto in LISP e i dati di esso rappresentati in LISP, ritorna come valore il risultato dell'applicazione del programma sui suoi dati. L'elemento fondamentale di questo approccio, al quale si sono in seguito ispirati gli studiosi del Vienna Definition Language, è che essendo sia il programma che i dati rappresentati in LISP risulta possibile scrivere un programma, la funzione APPLY, che opera su di essi. Altrettanto fondamentale è l'aver trattato un programma alla stregua di un dato.
Rispetto a queste ricerche l'elemento innovatore introdotto dal gruppo di Vienna è stato quello di usare lo stesso metalinguaggio, sia per la descrizione formale dei

Fig. 5 - Processo di compilazione di un linguaggio di programmazione



programmi, sia per la descrizione formale del loro interpretatore cioè dei dati e degli stati dell'interpretatore stesso nelle varie fasi dell'esecuzione.
È possibile a questo punto esaminare l'impatto della teoria dei linguaggi in termini di definizione sintattica e semantica sul processo di compilazione dei linguaggi programmativi ad alto livello.
Questo processo si articola nelle tre fasi di: *analisi lessicale* nella quale vengono isolati e identificati i componenti elementari costituenti il linguaggio (verbi, nomi, delimitatori, operatori, etc.); *analisi sintattica* durante la quale vengono riconosciuti i costrutti sintattici (espressioni aritmetiche, frasi di descrizione dei dati, frasi di comando, etc.); *generazione del codice* nella quale ad ogni costrutto sintattico viene associata una sequenza di istruzioni nel codice della macchina oggetto che riproducono gli effetti desiderati (v. fig. 5).
La descrizione sintattica di un linguaggio di programmazione assume quindi tutta la sua importanza quando si devono costruire le prime due fasi di un compilatore, l'ultima fase è quella più propriamente legata agli aspetti semantici del linguaggio sorgente.

5 - Futuri sviluppi

Come detto in precedenza, l'interesse attuale dei ricercatori si sta orientando soprattutto su quegli aspetti dei linguaggi relativi alla semantica ed alla pragmatica. Infatti per ciò che concerne la sintassi le ricerche tendono principalmente all'aumento dell'efficienza dei compilatori o a mettere in luce alcuni aspetti particolari di una teoria ormai consolidata (30).
Dove al contrario resta molto da fare è nel campo della semantica dei linguaggi e dei programmi; ricerche in questo senso potrebbero portare al raggiungimento di obiettivi ambiziosi, quali la possibilità di realizzare automaticamente un compilatore per un qualsiasi linguaggio, opportunamente descritto, per una qualsiasi macchina, anch'essa opportunamente descritta (compiler compiler) (31). Inoltre i problemi della correttezza dei programmi e della loro sintesi automatica sono anch'essi legati alle ricerche sulla semantica (32).
Per quel che concerne la pragmatica dei linguaggi, cioè le loro modalità d'uso per la stesura di programmi corretti e verificabili, si può affermare che questo aspetto ha sicuramente un notevole impatto sui costi di sviluppo e di gestione dei programmi stessi ed è, per questo, oggetto di notevoli studi. Quest'ultimo argomento, in particolare, può essere inquadrato in quella disciplina di più ampio spettro che si preoccupa della razionalizzazione ed ingegnerizzazione dell'attività di produzione e gestione dei programmi (software engineering), (33) almeno per ciò che concerne la progettazione di linguaggi di più facile impiego e le modalità per usarli e per insegnarne l'uso corretto.
Relativamente a probabili futuri sviluppi degli anni '80, il discorso appare complesso sebbene alcune tendenze già si avvertano. Per i linguaggi destinati alla grande produzione di complessi sistemi, come si è detto, si scorge chiaramente all'orizzonte la "terza generazione": infatti sia il Dipartimento della Difesa degli USA, che la Comunità Economica Europea, hanno varato piani per la definizione di linguaggi universali da adottare nei prossimi anni. Questi linguaggi, HOL – High Order Language ed ESL – European System Language (34), saranno derivati del PASCAL migliorati sia alla luce delle ricerche svolte negli ultimi anni, il PASCAL è del '69, sia tenendo conto di alcune caratteristiche tipiche dell'ambiente industriale in cui saranno utilizzati e non esplicitamente considerate nella definizione originale del PASCAL.
Un ultimo punto da considerare trattando dei futuri sviluppi dei linguaggi programmativi è quello dei linguaggi completamente non-procedurali ed in particolare la possibilità a medio termine della realizzazione di macchine programmabili direttamente dall'utente in termini descrittivi del problema.
È pensabile che si arrivi a precisare alla macchina le sole caratteristiche esterne dei programmi e non quelle interne; in realtà tale problema (programmazione automatica) è ancora lontano da una soluzione praticabile per ciò che concerne i soli aspetti funzionali dei programmi ed ancora da affrontare appieno per gli aspetti relativi alle loro prestazioni. Questi ultimi, se sono divenuti meno pressanti per via del minor costo della macchine, sono ancora importantissimi specialmente per i grossi sistemi e per i "packages", quindi per la maggior parte dei programmi prodotti. Qualora fosse risolto questo non semplice problema, resterebbe da considerare quanto segue.
Abbiamo già visto che linguaggi non-procedurali sono stati realizzati sotto pena di grosse limitazioni per ciò che concerne l'ampiezza delle problematiche affrontate; la questione è se sia possibile non pagare questo scotto.
In realtà, a parte problemi ancora insoluti relativi all'intelligenza artificiale, un aspetto importante che caratterizza la comunicazione umana è la assunzione di un contesto nel quale valgono le varie affermazioni; la sua conoscenza è indispensabile alla loro comprensione.
Nel caso dell'utilizzo di una macchina generale questo contesto dovrebbe essere ricostruito: lo sforzo di una tale ricostruzione sommato a quello di descrizione del problema, in generale, dipende solamente dalla complessità intrinseca del problema stesso. Ad esempio i linguaggi non-procedurali, di cui abbiamo precedentemente parlato, sottintendono un contesto molto particolare e questo conferisce molta potenza alle istruzioni.
Quindi la quantità di informazioni che debbono essere fornite ad una macchina è senz'altro maggiore di quella necessaria alla descrizione del problema in termini indipendenti dal particolare settore applicativo interessato; e almeno per le grandi applicazioni, è sicuramente maggiore di quanto l'utente finale sia in grado di dare senza l'aiuto di un intermediario esperto, cioè il programmatore applicativo.

Sembra pertanto che, se si raggiungerà nel breve termine una minore spesa di produzione e gestione dei programmi, questo avverrà specialmente attraverso la ingegnerizzazione della produzione di questi e nell'utilizzo di prodotti generali per più utenti.

6 - Bibliografia

(1) P. Wegner, *Programming Language, Information Structures and Machine Organization.* New York: Mc Graw-Hill, 1968.

(2) Ohlson M., *«System Design Consideration For Graphic Input Devices»,* IEEE, Computer Magazine, Vol. 11, n. 11, Nov. 1978.

(3) J. Von Neumann, *«The EDVAC Report»,* Computer from Pascal to Von Neumann, H. Goldstein, Ed. Princeton, NY Princeton University Press, 1972.

(4) M.V. Wilkes, D.J. Wheeler, and S. Gill, *The Preparation of Programs for a Digital Computer.* New York: addison Wesley, 1951 (revised Ed. 1957).

(5) *«FORTRAN vs. basic FORTRAN»* Comun. Ass. Comput. Mach., Oct. 1964.

(6) P. Naur, Ed., *«Report on the Algoritmic Language ALGOL 60»,* Commun. Ass. Comput. Mach. May 1960.

(7) COBOL 1961, *Revised Specifications for a Common Business Oriented Programming Language,* U.S. Gvt. Printing Office, 1961.

(8) G.M. Palermo *«Caratteristiche strutturali dei Linguaggi per il trattamento degli archivi»,* Estratto dal *Seminario sui Sistemi di reperimento e selezione automatica dell'informazione,* Accademia Nazionale dei Lincei, Roma, 17-21 Aprile 1972.

(9) J. Sammet, *Programming Languages, History and Fundamentals.* Englewood Cliffs, NJ: Prentice Hall, 1969.

(10) PL/I, *Current IBM System 360 Reference Manual,* (or Bates and Douglas), 2nd Ed. Englewood Cliffs, NJ: Prentice Hall, 1975.

(11) V. Wingaarden at al., *«Report On the Algorithmic Language ALGOL 68»,* Numer. Math., Feb. 1969, also revised report, Numer. Math., Feb. 1975.

(12) P. Wegner, *«Abstraction - a tool in the management of Complexity, in* Proc, 4th Texas Sump. Comput., Nov. 1975.

(13) C.A.R. Hoare and N. Wirth, *«An Axiomatic Definition of the Programming Language PASCAL»,* Acta Informatica, vol. 2, n. 4, 1973.

(14) B.H. Liskov, *«A note on CLU»,* Computation Structures group Memo 112, Nov. 1974.

(15) W. Wulf, R.L. London, and M. Shaw, *«Abstraction and Verification in ALPHARD, introduction to language and methodology»,* Carnegie Mellon Univ. Dept. Comp. Sci. Reg. June 1978

(16) C. Ghezzi, D. Mondriali, *«Specifiche e prove di proprietà dei programmi: lo stato dell'arte»,* Informatica, 8, N. 3, Luglio-Settembre 1978.

(17) O.J. Dahl, E.W. Dijkstra, C.A.R. Hoare, *Structured Programming,* Accademic Press, NY, (1972).

(18) J. McCarthy at al., *LISP 1.5 Programmers Manual.* Cambridge, MA MIT Press, 1965.

(19) R. Griswold, J. Poage, and I. Polonsky, *The SNOBOL 4 Programming Language Englewood Cliffs, NJ Prentice Hall, 1971.*

(20) *Comit Programmers' reference manual,* M.I.I. Research lab. of Electronics and the Computation centre, Cambridge, Mass. Nov 1961.

(21) K.E. Iverson, *A Programming Language* New York: Wiley, 1972.

(22) N. Chomsky, G.A. Miller, *Introduction to the formal Analysis of Natural Languages,* Handbook of Matematical Psychology, Vol. II NY: Wiley 1963.

(23) A.V. Aho, J.R. Ullman, *The theory of Parsing, Translation and Compiling* Englewood Cliffs, NY Prentice Hall, Vol. 1°, 1972, Vol. 2° 1973.

(24) Z. Manna, *Mathematical Theory of Computation,* McGraw.

(25) S. Strachey, «*Towards a formal semantics*», T.B. Steel editor.

(26) D. Scott, S. Strachey *«Towards a mathematical semantics for computer language»,* PRG6, Oxford Univ. Comp. Lab., 1971.

(27) D.E. Knuth, *«The sematics of Context free Languages»,* in Mathematical Systems theory, vol. II, n. 2, 1968.

(28) P. Lucas, *«On the formalization of Programming Languages: Early History and main approaches»,* in The Vienna Development Method: the Meta-Language, Lecture Notes in Computer Science, Springer Verlag, Berlin Heidelberg New York, n. 61, 1978.

(29) M. Wirth and H. Weber, *«Euler - A generalization of ALGOL and its formal definition»,* Commun. Ass. Comput. Mach., Jan. and Feb. 1966.

(30) J. Lewi, K. De Vlaeminck, J. Hens, M. Huybachts: *«The ELL (1) Parser Generator and the Error Recovery Mechanism»,* Acta Informatica, 10 N. 3, (1978)

(31) A.V. Aho, R. Sethi, *«How hard is compiler code generation?»* Automata Languages and Programming Lectures notes in computer science, ed. by G. Goos and J. Hartmanis, July 1977, Springer Verlag, NY.

(32) R.J. Waldinger *«Constructing programs automatically using theorem proving»,* PhD thesis, Carnegie Wallon Univ. May 1969.

(33) B.W. Boehm, Software Engineering, *IEEE Trans. on Software Eng.,* C. 25, 12, Dec. 1976

(34) *ESL Questionnaire,* Siemens, CII Honeywell Bull Jan. 1979.

La produttività di programmazione nello sviluppo del software

DOROTHY A. WALSH
Consulente EDP

1 - Introduzione

Malgrado i progressi delle tecnologie dell'elaborazione dati e l'introduzione di metodologie specifiche per l'attività di programmazione, la valutazione 'a priori', ovvero la stima, di quello che può significare in tempo, risorse ed altri oneri lo sviluppo del software, sembra ancora come fatta con la classica sfera di cristallo. E con una sfera annebbiata, a giudicare dal coro di lagnanze per mancato rispetto di tempi, costi ecc. che capita quasi regolarmente di sentire nel mondo dell'elaborazione dati. Nondimeno, coll'aumentare dei costi di produzione del software, diviene sempre più necessario disporre di criteri di stima che permettano di valutare e pianificare più affidabilmente gli oneri di produzione del software.

2 - Pseudo-metodi e miti

I motivi per cui la stima per il software è di fatto più un «predire la fortuna» che non un effettivo processo di misura sono complessi e vari. Del resto, se così non fosse, la soluzione del problema di fare stime realistiche verrebbe ad essere sostanzialmente affidata ad una diligente applicazione degli strumenti di misura per ottenere stime accurate quanto necessario. È invece la natura stessa del lavoro di «produrre software», con le numerose componenti tecniche e non tecniche implicate, che rende il compito di fare stime un'attività delicata e complessa. Modi di lavorare, linguaggi di programmazione impiegati, metodologie avanzate possono fortemente influenzare lo sviluppo del software così come lo possono, in modo altrettanto determinante, le numerose e diverse contingenze che si manifestano nell'ambiente di produzione in modo scarsamente prevedibile.

Se dunque il problema di formulare stime è per sua natura accidentato e difficile da impostare, a complicarlo ulteriormente si aggiungono errate consuetudini, sostenute più da una gratuita convinzione che le cose dovrebbero andare in un certo modo, che da sperimentazione effettiva quand'anche lacunosa. Una consuetudine molto diffusa, quanto fallace, è quella del «metodo della percentuale» o «del rapporto», secondo il quale il ciclo di sviluppo del software viene ripartito in fasi (quali disegno, minutazione, test, ecc.) ed una di esse stimata indipendentemente; dopodiché si risale all'onere totale, nonché a quello delle altre singole fasi, sulla base di una presunta relazione percentuale esistente tra esse. Nella letteratura — che possiamo ormai dire di ieri — sul «software management development», si ritrovano frequentemente semplicistici diagrammi a forma di torta divisa in settori più o meno grandi che stavano proporzionalmente ad indicare il 'quanto' impegno, del tutto, una certa attività del lavoro di produzione del software avrebbe richiesto (Fig. 1). In sostanza questo criterio dava per scontato — anche se tutt'altro che provato — che esistesse e fosse nota una fondamentale relazione invariante tra una fase e l'altra. Ne conseguiva che, noti ad esempio i costi di minutazione per un determinato lavoro, i costi di test del software prodotto potevano essere stimati in funzione dei primi.

Orbene, questa credenza ha portato a molte deludenti esperienze ed a sorprese specialmente quando 'a posteriori' si scopriva dove in realtà erano andate a finire le varie risorse, in ultima analisi economiche, investite nel progetto, in proporzione ben diverse da quelle presentate nel grafico a torta e per un ammontare complessivo ben distante da quello previsto.

Né si ottenevano risultati migliori quando, noti i costi di minutazione e test di un certo progetto già concluso, si usavano gli stessi per stimare i tempi di test di un altro progetto che presentasse gli stessi costi di minutazione del precedente. È invece esperienza di tutti quelli che lavorano nel software l'essersi imbattuti in programmi che pur avendo presentato lo stesso tempo di minutazione hanno poi richiesto tempi di test considerevolmente diversi.

La Tabella 1 mostra come è risultato speso il tempo in vari progetti software. È interessante notare come la sola correlazione emergente è tra disegno e test — e non tra minutazione e test — nel senso che tanto maggiore è stato l'impegno dedicato al disegno tanto minore è stato quello richiesto per il test.

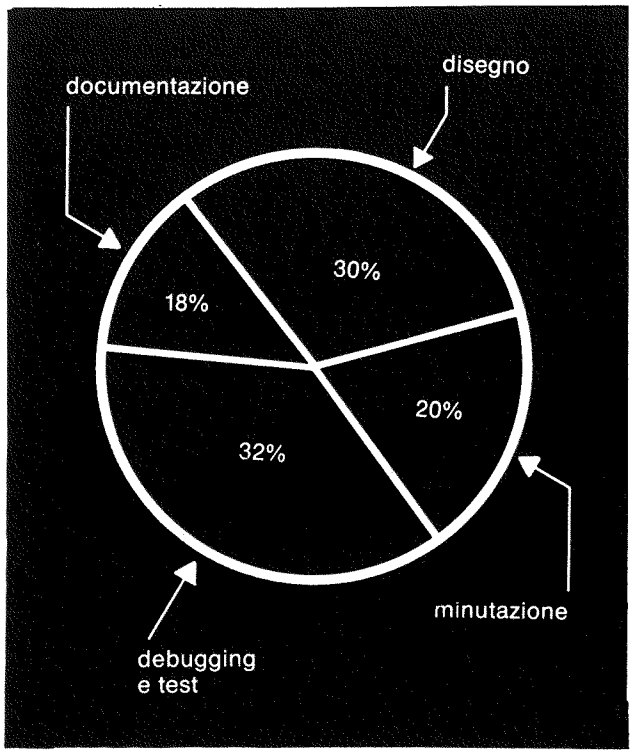


Fig. 1. Un tipico diagramma «a torta» — molto comune nella letteratura di ieri sulla conduzione di progetti software — che intendeva mostrare presunte relazioni quantitative esistenti tra le principali attività dello sviluppo del prodotto software. Si noti che il lavoro di documentazione era visto come un'attività indipendente e successiva al completamento del software, anziché — come oggi consentono le tecniche top-down e l'uso dei linguaggi ad alto livello — integrata durante tutta l'attività di produzione del software.

3 - La produttività di programmazione

Quello che a tutt'oggi è risultato essere il parametro più affidabile per far stime di sviluppo del software è quello della «produttività di programmazione». Nondimeno da studi sull'argomento s'è venuta formulando una definizione abbastanza comune di ciò che debba intedersi per produttività di programmazione.

Questo parametro si esprime in funzione del numero di «linee di programma» che compongono un sistema di software finito (Delivered Source Lines of Code: DSL) e del tempo complessivo per la sua realizzazione; cioè dal disegno alla consegna per l'uso del sistema finito. Le linee di programma prese in considerazione includono le definizioni di campi e costanti, i commenti e le istruzioni di programma vere e proprie, contate a sistema completato, cioè pronto per l'uso da parte dell'utenza finale. Il tempo richiesto per produrre il sistema finale va computato dal momento in cui si è cominciato il disegno del software da realizzare al momento della consegna del prodotto per l'uso. La misura della produttività può quindi esprimersi come:

$$P = \frac{DSL}{UT \times NP}$$

dove:
P = Produttività
DSL = numero complessivo di linee di programma del software completato
UT = unità di tempo assunta (giorni; mesi ecc.)
NP = numero di persone impegnate nel progetto

È ovvio che il totale delle linee di programma di un sistema completato non può non dipendere dal tipo di linguaggio di programmazione usato. A parità di funzioni espletate, un programma scritto con un linguaggio ad alto livello presenta infatti un numero di linee di programma minori di quelle che lo stesso programma presenta se scritto con un linguaggio di tipo assemblativo. Va quindi tenuto conto che i dati ricavati da ricerche sulla produttività di programmazione sono relativi ai particolari tipi di linguaggi di programmazione e vanno quindi usati accortamente nel caso li si voglia applicare a progetti software scritti in linguaggi diversi. Un'ulteriore particolare da tener presente è quello relativo al tipo

Tabella 1. I dati mostrano come sono distribuiti gli oneri (in anni/uomo) di realizzazione di noti progetti software. Questi dati sono stati ricavati a progetto completato; pertanto mostrano il tempo effettivamente speso nelle varie fasi (come percentuale di quello totale) e non quello stimato (1).

Progetto	analisi e disegno (%)	minutazione e debugging (%)	test e validazione (%)
SAGE (Semi-Automatic Ground Environment)	39	14	47
NTDS (Naval Tactical Data System)	30	20	50
SETE (Secretariat for Electronic Test Equipment)	35	17	48
	32	24	44
GEMINI	36	17	47
	33	17	50

di software, che in linea di massima può distinguersi in software di sistema e software applicativo.

4 - Alcuni dati di produttività di programmazione

I due studi principali che riportano la produttività in numero di linee di programma nell'unità di tempo presentano risultati coerenti tra loro. In entrambi gli studi sono considerati tanto progetti richiedenti limitati impegni di risorse umane (meno di 25 programmatori) quanto grossi impegni (più di 25 programmatori).
Nella tabella 2 e 3 sono presentati i dati ricavati dai due studi menzionati. I risultati della tabella 2, pubblicati nel 1974 (2), si riferiscono a progetti software portati a termine nei precedenti 8 anni e scritti in linguaggio assemblativo.
La Tabella 3 presenta dati, pubblicati per la prima volta nel 1977 (4), che si riferiscono a sistemi software scritti in COBOL e PL/1. Sono considerati sistemi piccoli e grandi, comunque già completati all'epoca della pubblicazione degli studi in questione.
I dati delle tabelle sono di origine sperimentale e ricavati 'a posteriori', cioè a progetto software concluso. Essi sono tuttavia utili per essere usati come base per formulare stime, almeno nell'ambito di sistemi software di caratteristiche similari a quelle dai quali sono state ricavate tali misure di produttività.

5 - Effetti particolari negli ambienti di sviluppo del software

Dai dati delle tabelle si rilevano variazioni anche notevoli dei valori di produttività quand'anche misurati su lavori di produzione del software di comparabile complessità. Orbene, dette differenze non possono essere semplicemente attribuite al linguaggio di programmazione usato. Un gruppo di risultati si riferisce infatti a linguaggi assemblativi di stesse caratteristiche generali, mentre l'altro si riferisce a programmi COBOL. Nè a spiegare queste differenze è valida l'ipotesi della eventuale particolare abilità di taluni programmatori; trattandosi infatti per la maggior parte di sperimentazione condotta con gruppi numerosi (più di 25 programmatori), possono essere ritenuti gruppi omogenei per ovvi effetti compensativi.
Si sono dunque cercati i motivi che determinavano queste differenze. Una ditta inglese produttrice di elaboratori ha trovato che le sue stime erano errate di un fattore 2 perché i programmatori venivano in effetti a spendere solo metà del tempo di calendario in lavoro produttivo (2).
Una ditta americana ha trovato che le interazioni necessarie tra i membri dei gruppi di lavoro riduceva la produttività anche drasticamente — in un caso la riduzione fu dalle normali 5000 linee di programma per anno a 1500 linee di programma all'anno; la causa di questo essendo la gran perdita di tempo in comunicazioni, meeting, ecc. (2).
Più recentemente, uno studio mirante a determinare i fattori contingenti che in un contesto di programmazione influenzano la produttività — sia in senso positivo che negativo — ha presentato risultati significativi (4). Alcuni di questi risultati sono raccolti nelle tabelle 4 e 5.
Anche questi dati, di origine sperimentale, sono utili per aggiustare le proprie stime una volta che si possano riconoscere condizioni comuni a quelle in cui essi sono stati determinati.

Tabella 2. Valori di produttività per tipo di progetto software scritto in linguaggio assemblativo. Si nota un prevedibile decremento della produttività in funzione della complessità propria del tipo di software.

Tipo di software	Linee di programma per anno/uomo
Sistemi operativi, Programmi supervisori	da 600 a 800
Raccolta dati	da 2000 a 3000
Applicazioni ordinarie	da 5000 a 6000

Numero complessivo di linee di programma del sistema software	Linee di programma per anno/uomo
2.700	15.000
3.000	11.800
60.000	3.500
75.000	3.900
140.000	2.600
910.000	2.300

Tabella 3. Valori di produttività misurati in sistemi software scritti in COBOL in funzione della loro dimensione in linee di programma. I dati sono stati ricavati da consuntivi espressi in linee di programma per giorni/uomo (3) e convertiti in linee di programma per anni/uomo, assumendo un anno/uomo uguale a 200 giorni/uomo. Si noti come i piccoli progetti software presentano un valore di produttività ben maggiore dei grandi progetti software.

6 - L'uso della produttività come parametro di stima

La sostanziale costanza delle misure della produttività di programmazione, è, come si è detto, un buon punto di partenza per formulare stime degli impegni necessari per la realizzazione di un progetto software. Una stima è tuttavia un'operazione più complessa e sempre correlata al singolo caso, e deve tenere conto, in modo globale, di una varietà di componenti, non escluse possibili circostanze contingenti che si possono probabilisticamente manifestare nel contesto in cui il software sarà prodotto.
Il primo passo per fare una stima realistica del tempo e delle altre risorse per la realizzazione di un sistema software sta dunque nell'ottenere un'affidabile valutazione del numero di linee di programma di cui sarà costituito il sistema software finale. Questa valutazione può essere fatta da personale tecnico specialistico, possibilmente come media delle valutazioni condotte da un gruppo di tecnici di varia esperienza, o da consulenti particolar-

Circostanza	Effetto negativo sulla produttività (espresso in % del livello normale statistico assunto = 100)
• Complessità di disegno	69
• Numerose modifiche richieste a progetto avviato	65
• Vincoli di disegno molto restrittivi	58
• Vincoli molto stretti nei tempi di risposta	54
• Nuovi linguaggi di programmazione	54
• Nuovo elaboratore	54
• Scarsa esperienza del personale	51
• Interfacce complesse tra i moduli costituenti il sistema software	42

Tabella 4. I dati in tabella sono ricavati dai risultati di uno specifico ed ampio programma di misure sul software condotto alla Federal Systems Division dell'IBM (Gaithersburg, Maryland). Le circostanze elencate hanno sperimentalmente dimostrato una sicura correlazione negativa con la produttività.

mente competenti nello specifico settore applicativo. Elementi di ausilio per dette valutazioni sono il numero di linee di programma di sistemi software già noti per applicazioni similari, o di parti componenti, la maggiore o minore specializzazione per la particolare applicazione del linguaggio di programmazione da usare, la disponibilità di determinato software di base, ecc.

Una volta preparata la fondamentale valutazione del numero di linee di programma prevedibili per il sistema software finale, questa va corretta per mettere in conto eventuali circostanze che possono manifestarsi nell'ambiente di produzione del software. Si supponga, a titolo di esempio, che si preveda che le variazioni e modifiche al disegno iniziale possano essere alte, e che l'esperienza con il linguaggio di programmazione da usare sia poca. Orbene la tabella 4 mostra che la prima circostanza può ridurre il livello di produttività al 65% rispetto a quello ritenuto normale, assunto come 100, e la seconda ai 54%.

Lo studio che ha presentato i dati in questione, si limitava a fornire l'effetto sulla produttività dei vari fattori, presi singolarmente, ma non correlava tra loro gli effetti dei detti fattori, che per loro natura possono presentarsi tanto singolarmente che variamente combinati nella realtà dei singoli casi. Pertanto non è a rigore lecito estrapolare dai dati in questione — se non con una finalità puramente orientativa — l'effetto quantitativo risultante dalla presenza dei due fattori. Ciò premesso, supponendo tuttavia di poter assumere il valore medio dei due dati, si calcola un effetto di degradazione della produttività che la riduce a circa il 60% del normale. Nella casistica della tabella 3, applicandó per esempio quest'ultimo coefficiente di correzione al valore di produttività per un progetto software di 9 10.000 linee di programma, il previsto valore di produttività di 2.300 linee di programma per anno/uomo in condizioni normali viene a ridursi a circa 1.310 linee di programma per anno/uomo.

Messi in conto i fattori negativi, si può poi apportare un'ulteriore correzione in funzione di eventuali circostanze favorevoli. Si considerino ad esempio circostanze favorevoli quali la disponibilità di personale molto qualificato e l'impiego della programmazione strutturata.

Dalla tabella 5 si nota che queste due circostanze possono rispettivamente portare ad un aumento della produttività del 182% e del 178% del normale.

Anche qui vale la considerazione che trattandosi di dati rilevati sperimentalmente e singolarmente, l'effetto della presenza simultanea dei due fattori indipendenti dovrebbe essere anch'esso misurato sperimentalmente, non essendo affatto a priori supponibile che sia uguale alla risultante algebrica dei dati relativi alle due circostanze prese singolarmente. Sempre in un'ottica semplicemente orientativa, si potrebbe considerare un'unico fattore di incremento pari alla media dei due, ovvero un fattore di incremento della produttività del 180%.

Tuttavia i dati e le considerazioni presentate vanno applicate col dovuto buon senso, e non meccanicamente. Si osservi per esempio che non si peccherebbe certo di ottimismo se nel caso in questione si considerasse come assente — anche se non lo fosse — la circostanza positiva della programmazione strutturata, e si considerasse il solo effetto favorevole portato dalla disponibilità di personale molto qualificato. Si comprende allora che nel caso in esempio il reale fattore di incremento della produttività è di almeno 182%.

7 - Anni uomo ed anni calendario

I valori di produttività a cui si è così arrivati possono ora essere utilizzati per determinare il tempo complessico (in anni/uomo) necessario per la esecuzione del lavoro.

Se il numero di linee di programma stimato è per esem-

Tabella 5. I dati in tabella hanno la stessa origine di quelli in tabella 4. Si osservi come la sperimentazione dimostri che i benefici della programmazione strutturata sono tutt'altro che opinabili o quantitativamente trascurabili. È anche interessante rilevare l'effetto benefico del coinvolgimento più ampio del personale; effetto che non deve attribuirsi solo ad una più completa conoscenza del lavoro da svolgere, ma anche all'implicita motivazione che deriva dall'essere coinvolti nel lavoro in veste più ampia e qualificata.

Circostanza	Effetto positivo sulla produttività (espresso in % del livello normale statistico assunto = 100)
• Personale di progetto molto qualificato	182
• Ampio uso di tecniche di programmazione strutturata	178
• Personale destinato alla programmazione molto esperto	171
• Alta partecipazione del personale destinato alla programmazione anche al lavoro di disegno	161
• Impiego di tecniche top-down di sviluppo del software	135
• Ampio uso di tecniche di 'code inspection' e 'design inspection'	113

pio 200,000, e il valore di produttività calcolato è di 4500, allora il numero di anni/uomo necessari sarebbe di 200,000/4500 ovvero 44,4 anni/uomo. Il valore risultante di anni/uomo deve ora essere convertito in una forma che esprima il numero di persone necessarie a soddisfarla. È intuibile che ciò non può essere fatto in modo semplice e lineare. A fronte cioè di un valore di 44.4 anni/uomo, non si può semplicisticamente affermare che impegnando 45 persone il lavoro potrà essere concluso in un anno. Se è infatti vero che certe attività possono essere condotte in parallelo, come la stesura dei programmi ed il test eseguito con la tecnica degli scheletri di programma, le revisioni del disegno e la pianificazione dei test ecc., altre attività devono invece essere necessariamente eseguite in serie. Quindi, la conversione da anni/uomo al numero di persone necessarie ed al conseguente tempo di calendario deve essere fatta tenendo conto della varie fasi dello sviluppo software e del peso di ciascuna nel lavoro complessivo.
Per quest'ultimo compito possono seguirsi varie vie. Il primo lavoro da stimare è quello del disegno dettagliato. Il tempo necessario per il disegno dettagliato deve essere stimato in anni/uomo, o nell'unità di misura più opportuna (mesi/settimane, giorni/uomo), dal personale tecnico competente. Come elemento di controllo della stima del disegno si può tener conto che questo deve essere circa il 40% dell'impegno totale del lavoro di realizzazione del progetto.
Considerando allora circa 44,4 anni/uomo come totale e destinando il 40% per il disegno, risulta per il disegno 17,7 anni/uomo. A questo punto è opportuno prendere in considerazione la complessità, ovvero il livello di modularità, del software da sviluppare al fine di stabilire il numero di componenti dell'equipe di lavoro. È infatti principalmente in funzione della complessità e modularità che si può decidere, se, per esempio, impegnare 20 persone per un anno o 10 persone per due anni.

8 - Distribuzione di impegni di lavoro ed estrapolazione delle stime

Considerare che circa il 40% dell'impegno totale è assorbito dal disegno non implica automaticamente che una volta nota l'entità assoluta di detto impegno, la corrispondente minutazione dovrà essere del 20%, e così via percentualmente per le altre attività, fino a giungere al valore di impegno totale risultante. Ciò che si intende fare è invece distribuire il tempo totale stimato già predefinito, allo scopo di pianificare adeguate risorse per rientrare in esso. Si tenga conto infatti che il 40% dell'impegno complessivo non è il 40% del tempo calendario complessivo. È a questo punto che si presenta l'importante problema del numero e delle caratteristiche dei componenti l'equipe di lavoro. Una cautela da tener presente è, come già detto, di far sì che il personale impiegato per l'esecuzione del disegno dettagliato sia, finché è possibile, lo stesso personale che sarà poi impegnato per la successiva implementazione.
Per ciascuna attività, quindi, l'impegno totale dovrebbe essere ripartito tra il personale ed il tempo necessario, per calcolare le esigenze di risorse così come il tempo di calendario.

9 - Le stime e la realtà

Fatte le stime ed iniziato il lavoro di sviluppo del software possono facilmente presentarsi situazioni non previste, e difficilmente quantizzabili in maniera preventiva (assenze del personale per malattia, dimissioni, ritardata disponibilità delle macchine su cui provare i programmi, ecc.), che inficiano immediatamente, in maniera raramente del tutto recuperabile, i tempi ed i costi pianificati. In casi del genere, nell'adottare degli interventi compensativi, bisogna aver cura di valutare se una azione compensativa effettivamente porta benefici reali, e di non violare le relazioni di serialità e parallelismo che legano determinate attività (5). Aggiungere o rimpiazzare del personale in un progetto già in fase avanzata di sviluppo comporta fatalmente un aumento delle comunicazioni necessarie; inoltre, talvolta, può anche essere necessario un tempo preliminare di addestramento per i nuovi venuti; tempo ovviamente addizionale e non previsto nelle stime originarie. Può anzi manifestarsi in certe situazioni l'effetto, apparentemente paradossale, che tanto maggiore è il numero delle nuove persone inserite a progetto iniziato con finalità compensativa e di recupero, tanto maggiore è il tempo che si finisce per sprecare in attività extra, non pianificate. È quindi importante aver cura di valutare l'effetto globale di ogni azione intrapresa per rimediare ad una situazione di ritardo che nasce imprevista durante il lavoro. Si consideri per esempio il caso molto comune di un ritardo in fase di minutazione dei programmi perché un programmatore si è ammalato ed è assente per un mese. Se era previsto che il suo lavoro si dovesse completare in due mesi, si potrebbe essere tenuti ad inserire una nuova persona per fare quel lavoro. La nuova persona richiederà, nella migliore delle ipotesi, almeno una settimana per ambientarsi ed entrare nello spirito del lavoro. In questa settimana il nuovo venuto richiederà la collaborazione di almeno un'altra persona che lo metta al corrente su ciò che deve fare. Ciò significa che, tra una cosa e l'altra, passa un mese/uomo in attività che non contribuiscono all'effettiva produzione del software e che non comportano quindi il recupero del mese di ritardo subito, ottenendosi di fatto, nella migliore delle ipotesi, lo stesso risultato pratico finale che si sarebbe ottenuto attendendo il rientro del programmatore ammalato assente.

10 - Conclusioni

L'arte di fare stime affidabili per la realizzazione di progetti software sta oggi sempre più spostandosi da un piano puramente empirico per assumere caratteristiche di rigore scientifico. Ciò è dovuto soprattutto alla sufficiente sperimentazione che ha permesso di riconoscere nel parametro «produttività di programmazione» quello tra i più affidabili per dedurre e quantizzare i vari impegni per lo sviluppo di un progetto software. Per produttività di programmazione si intende il numero complessivo di «linee di programma» scritte da un programmatore nell'unità di tempo (giorno, mese, ecc. secondo i casi), misurate a sistema software finale completo e funzionante. Per «linea di programma», tutti gli studi sulla sperimentazione in merito intendono non solo le istruzioni effettive di programma, ma anche le definizioni di campi e costanti e le linee di frasi di commento.

L'interesse per il parametro produttività di programmazione sta nel fatto che, da disparate misure e sperimentazioni, esso risulta essere quello che presenta una maggior coerenza e costanza di valori, per cui meglio si presta ad essere assunto come base per formulare stime.
Una volta noto il numero complessivo di linee di programma che andranno a costituire un determinato sistema software, ed il coefficiente di produttività di programmazione per quel tipo di software e per il linguaggio di programmazione usato, si può risalire al lavoro in anni/uomo necessario per realizzare il sistema di software, e quindi convertire questa valutazione in tempo e personale richiesto.
Come tutte le misure basate su valutazioni essenzialmente sperimentali, è tuttavia necessario aggiustare ed affinare i valori stimati sulla base della miglior conoscenza dello specifico ambiente di sviluppo del software e sulla base di possibili circostanze contingenti ipotizzabili.

11 - Bibliografia

(1) B.W. Boehm, «Software and its impact: a quantitative assessment», Datamation, 19, No. 5, 48-59, May 1973.
(2) F. Brooks, « The mythical man month», Addison-Wesley
(3) J.R. Johnson, «A working measure of productivity», Datamation 23, No. 2, 106-112, February 1977.
(4) C.E. Walston adn C. P. Felix, «A method of programming measure- and estimation», IBM Systems Journal, 16, No. 1,54-73, 1977.
(5) D.A. Walsh, «Planning: the key to on time software», Management Informatics, 2, No. 5, 221-225, October 1973.

La automazione dell'ufficio

GIULIO OCCHINI
Honeywell Information Systems Italia
Milano

1 - Introduzione

In questo articolo si esamineranno alcune possibilità di impiego della tecnologia informatica nell'area delle attività di ufficio, cioè di tutte quelle funzioni di carattere amministrativo-logistico che fanno parte del lavoro impiegatizio e che sono rivolte al trattamento e alla rice-trasmissione delle informazioni indipendentemente dal loro contenuto e dalla loro rappresentazione (digitale, vocale, grafica o testuale).

La concezione dell'ufficio automatizzato può dirsi avanzata non tanto perché adotti tecniche di avanguardia non ancora sufficientemente collaudate, quanto perché rappresenta la sintesi (fig. 1) di sviluppi sinora mantenutisi separati nelle aree:

– della elaborazione dati
– della trasmissione digitale
– delle apparecchiature di ufficio per la preparazione, la produzione e la archiviazione di documenti.

Va però anche sottolineato, anticipando quanto diremo al punto 6, come già oggi esistano sistemi che operano secondo i criteri dell'ufficio automatizzato e che tale integrazione stanno già sperimentando; il futuro che in questo capitolo si prospetta è quindi relativamente vicino, tanto da poterne sin d'ora delineare con chiarezza i contorni.

Perciò preferiamo parlare di «ufficio di domani» anziché, come spesso avviene, di «ufficio del futuro».

La caratteristica forse più importante dell'ufficio di domani è che si tratta della prima applicazione informatica veramente «di massa» nel senso che è destinata a una larghissima diffusione sia in senso estensivo, come numero di aziende candidate ad adottarla, sia in senso intensivo, come percentuale di persone che, all'interno delle aziende, ne verranno direttamente coinvolte.

La conseguenza è che la transizione dalla situazione attuale all'ufficio di domani comporterà certamente riflessi di carattere organizzativo e sociale che non sono a priori anticipabili in tutta la loro dimensione; questi, e non quelli tecnologici, saranno gli aspetti destinati a controllare la velocità del processo di trasformazione.

2 - La struttura del sistema

L'ufficio di domani può definirsi come un insieme di tecnologie, procedure di lavoro e persone, queste ultime con mansioni diverse e diversi livelli di responsabilità, finalizzato a produrre, trattare e rendere disponibili le informazioni che servono là dove e quando servono nell'ambito della organizzazione; e questo con una drastica riduzione dei compiti esecutivi e dei tempi di servizio rispetto all'ufficio di oggi.

Da questa definizione derivano logicamente alcune caratteristiche generali dell'ufficio di domani:

– l'ufficio di domani va concepito come un sistema integrato di servizi ad alto livello di transazionalità e non come un insieme di funzioni diverse che possano svolgersi a tempi prefissati e di apparecchiature particolari specializzate per funzione

– la sua gamma di impiego deve estendersi dalla fascia delle attività esecutive a quella delle funzioni direttive e professionali

– la implementazione dell'ufficio di domani ha come prerequisito una approfondita analisi:
. delle attività operative
. dei processi decisionali
. delle necessità e delle forme di comunicazione
. degli aspetti comportamentali a livello individuale e organizzativo

– i servizi dell'ufficio di domani dovranno estendersi alle aree:
. della elaborazione della parola, dei testi e del disegno (world/text processing e graphic processing) oltre che delle informazioni (data processing)
. della archiviazione e del reperimento dei dati, testi e documenti
. della «posta elettronica» (electronic mail) intesa come possibilità di memorizzazione ed interscambio continuo e in linea di messaggi e documenti e, in prospettiva, come possibilità di tenere riunioni a distanza (tele conferencing)

– la interfaccia attraverso cui tali servizi saranno resi di-

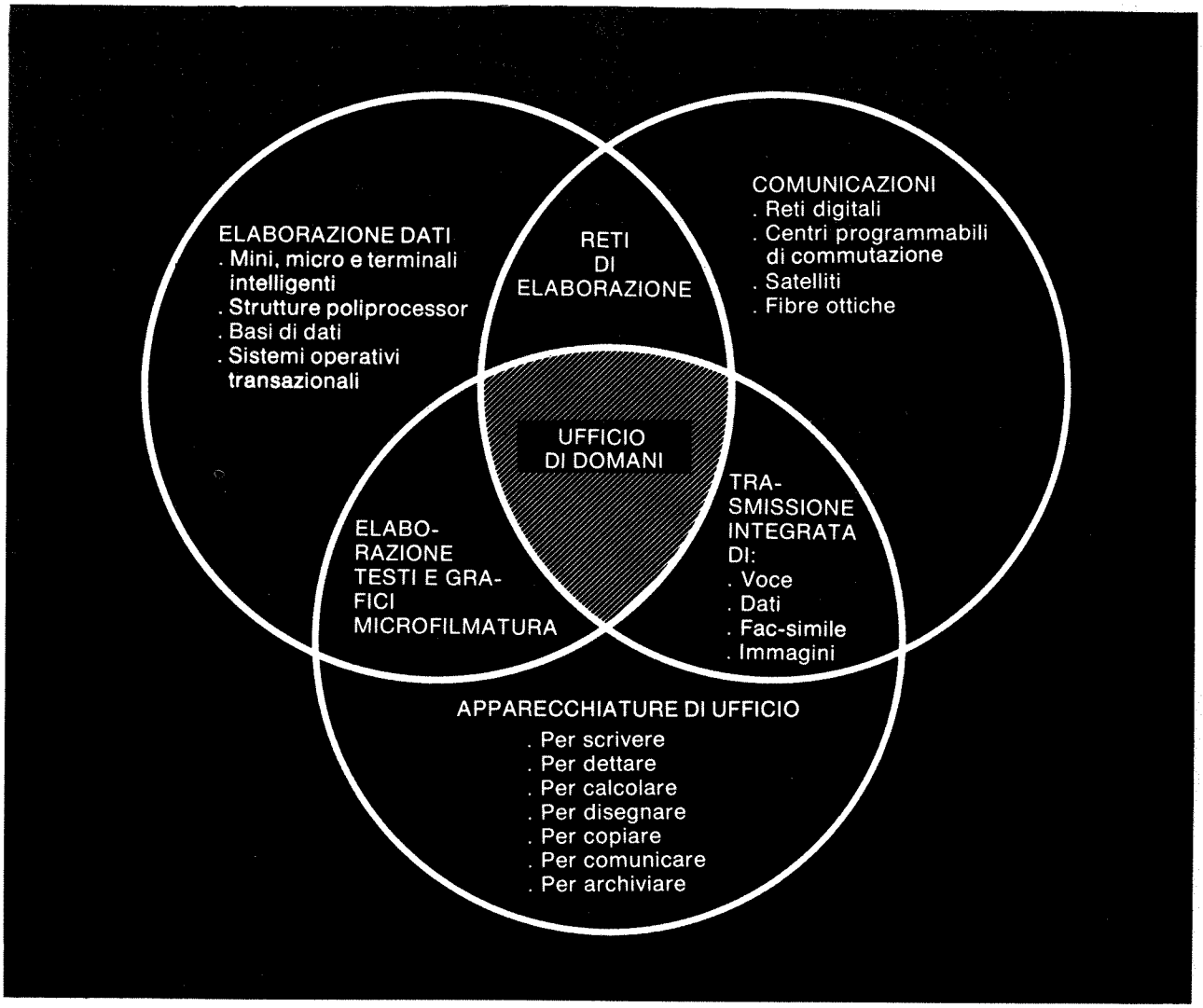


Fig. 1 - L'ufficio di domani come convergenza di sviluppi tecnologici diversi

sponibili, dovrà essere unificata e rivolta a personale privo di qualsiasi esperienza di informatica; in altri termini, le funzionalità del sistema dovranno rendersi del tutto trasparenti rispetto all'utilizzatore.

Dal punto di vista architettonico, la struttura del sistema dovrà essere di tipo distribuito con terminali generalmente costituiti da telefono, tastiera scrivente e schermo di visualizzazione in grado di trattare indifferentemente voce, testi, dati e grafici e dislocati nelle varie «stazioni di lavoro» dell'ufficio; tali terminali si collegheranno, a grappolo, a nodi di elaborazione locale (Fig. 2), e questi tra loro e con elaboratori di grandi dimensioni o comunque multifunzionali; la connessione sarà attuata tramite reti di comunicazione di tipo interconnesso.

Le ragioni per cui, sia pure con tutti gli adattamenti che si renderanno necessari per commisurarlo alle dimensioni della singola azienda, questo si ritiene debba essere lo schema sistemistico di riferimento, derivano ancora dagli obiettivi generali posti all'ufficio di domani.

Se la base informativa del sistema dovrà estendersi agli archivi documentari, le funzioni di gestione richiederanno una potenza elaborativa superiore di un ordine di grandezza a quella caratteristica degli attuali sistemi EDP; un analogo salto di prestazioni si renderà necessario per far fronte al volume di transazioni che le attività di ufficio genereranno; soltanto le soluzioni di tipo distribuito si prospettano oggi capaci di assorbire simili carichi di lavoro in modo efficiente e soprattutto di sapersi adattare dinamicamente ad una loro rapida crescita nel tempo continuando a garantire quella affidabilità di servizio che la applicazione richiede.

Analoga è anche la giustificazione della necessaria presenza, nel sistema, di elaboratori multifunzionali chiamati ad esercitare compiti di supervisione e controllo sulla rete e sulla base delle informazioni, nonché a garantire quella protezione e sicurezza di impiego che costituiscono un aspetto cruciale dell'ufficio di domani.

I nodi di automazione locale, ciascuno dei quali può comprendere uno o più elaboratori, dovranno essere abilitati sia al trattamento dei testi e dei grafici che a quello dei dati, e gli archivi che gestiscono dovranno fare parte integrante della base di informazioni del sistema complessivo.

Come si rileva dalla fig. 2 oltre ai terminali caratteristici delle stazioni di lavoro, al sistema di elaborazione locale si collegano anche altri tipi di apparecchiature; di tutte daremo nel seguito alcune caratteristiche rapportandole allo stato attuale dell'arte tecnologica.

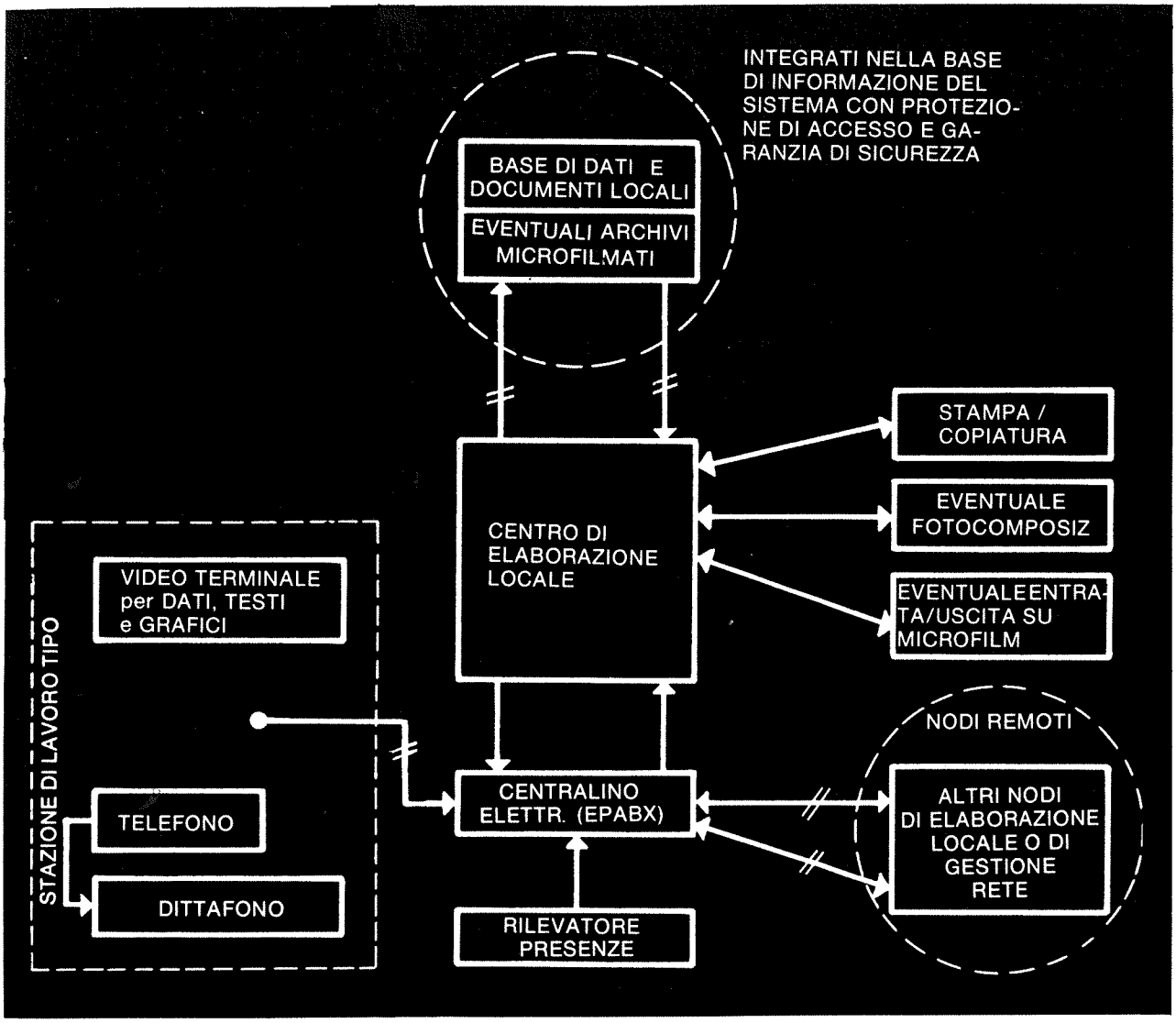


Fig. 2 - Nodo di automazione ufficio

2.1 - Commutazione elettronica privata

Sistemi di questo genere (centralini privati elettronici o, in inglese, EPABX da Electronic Private Automatic Branch Exchange) basati su minielaboratori, vengono già oggi offerti anche sul mercato italiano con interfaccia predisposta per il collegamento alla rete di comunicazione esterna (in Italia, quella telefonica pubblica gestita dalla SIP).

Il vantaggio di queste apparecchiature, rispetto ai normali centralini telefonici, deriva dalla loro flessibilità di programmazione e dalla capacità di memorizzazione di cui dispongono; si rende così possibile la esecuzione di tutta una serie di servizi amministrativi associati al traffico di comunicazione che il centralino normale non è in grado di fornire.

In particolare si possono citare:

— la integrazione della voce e dei dati sulla stessa linea che di volta in volta viene utilizzata dal telefono o dal terminale di ufficio

— la possibilità di accantonare temporaneamente comunicazioni provenienti dall'esterno e che non possono immediatamente essere trasmesse al destinatario

— la possibilità di raccogliere dati per successive elaborazioni (per esempio, dati di presenza del personale negli uffici o informazioni sul traffico di comunicazioni e sulla allocazione dei costi).

— la memorizzazione centralizzata di testi registrati su nastro attraverso i dittafoni degli uffici collegati, per la loro trascrizione successiva

— la possibilità di interrogare per via telefonica archivi o basi di dati ottenendone una risposta audio (touch-tone)

Il costo di queste apparecchiature, di cui in fig. 3 è rappresentato uno schema generale, è oggi, a parità di numero di linee, circa doppio di quello dei tradizionali centralini telefonici, ma la molteplicità degli impieghi cui si prestano li hanno già resi convenienti in molte situazioni.

È facile previsione anticipare una loro riduzione di prezzo nei prossimi anni che li renda competitivi e convenienti a tutti gli effetti.

2.2 - Trattamento dei documenti

Lo schema procedurale tipico che si segue nell'ufficio per la stesura e la diffusione di un documento è illustrato nella fig. 4; in questo schema sono messe in particolare evidenza le iterazioni che hanno luogo nella fase di preparazione

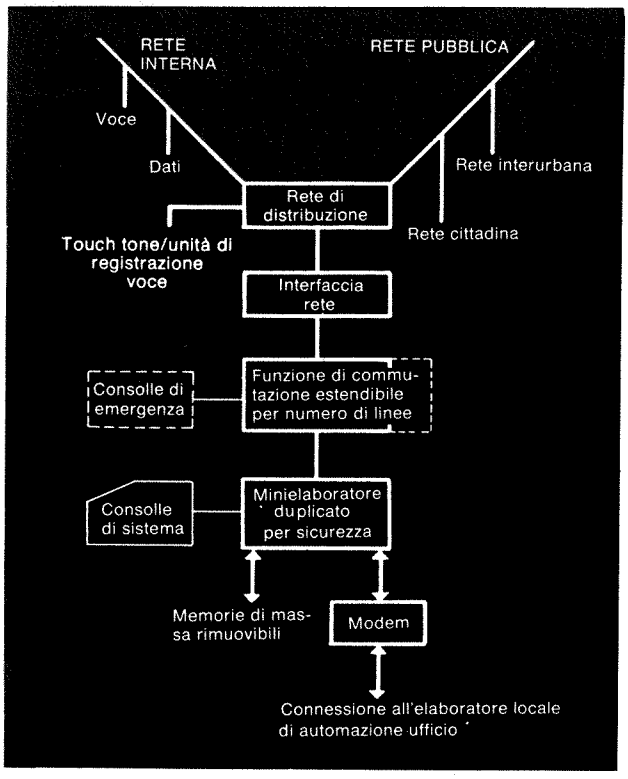


Fig. 3 - Schema funzionale di EPABX (centrale interna di commutazione elettronica)

e approvazione ed è appunto in questo ambito che si inseriscono i terminali di elaborazione testi.
Con tali terminali, il testo manoscritto o registrato su nastro per telefono (tramite EPABX) viene introdotto e trasmesso all'elaboratore su linea interna (se nello stesso edificio) o esterna (commutata o diretta).
Questa prima bozza viene quindi visualizzata sul terminale ed è sulla sua rappresentazione visiva che avvengono le correzioni, le modifiche e gli eventuali inserimenti da altri documenti memorizzati.
Una volta finalizzato, il documento può essere memorizzato e quindi, se va riprodotto su carta, automaticamente composto e riprodotto.
Il costo medio attuale della stazione indipendente di preparazione testi si aggira sui 6-7 Milioni di lire se costituita da tastiera e stampante e sugli 11 Milioni se dotata anche di video; i costi di collegamento e di elaborazione si può ritenere che incrementino queste cifre rispettivamente di un 10% e di un 5% nel caso il numero di stazioni collegate sia sufficientemente alto (15-20).
Tenendo conto della possibilità di centralizzare la funzione di stampa offerta dalla soluzione con elaboratore, si verifica agevolmente come, a parità di funzioni al terminale, già oggi la soluzione in linea risulti conveniente anche in termini puramente economici.
La funzione di riproduzione su carta, se necessaria, è delegata nell'ufficio di domani a una apparecchiatura detta convenzionalmente copiatrice intelligente; con questo nome si individua un processo di riproduzione di massa e ad alta qualità di stampa, di testi o immagini partendo sia da copia campione che da informazioni codificate su supporto magnetico o trasmesse in modo digitale per linea di comunicazione.
La struttura tecnologica dell'apparecchiatura può derivare dal tradizionale processo di fotocopiatura elettrostatica o da quello di composizione dei caratteri tramite leaser o getto di inchiostro; in tutti i casi testi e grafici verranno trattati in modo unificato. Esempi di queste apparecchiature sono già oggi disponibili sul mercato.
Alla stessa categoria appartengono concettualmente anche le fotocompositrici che troveranno un impiego là dove sono necessarie funzioni particolari di edizione (per esempio, più pagine di origine assiemate, rimpicciolite e concentrate su una unica pagina fotocomposta).
Per quanto infine riguarda il trattamento delle parti grafiche del documento, si ricorda come già oggi siano disponibili apparecchiature in grado di far fronte a qualunque necessità; in particolare, se le esigenze, come generalmente avviene, si limitano al tracciamento di istogrammi o di semplici curve di andamento, lo stesso terminale di composizione testi, che è destinato a coincidere con il classico terminale video EDP, può essere del tutto adeguato.
A completamento di queste considerazioni sulla preparazione dei documenti, va sottolineato come il fatto che si tratti di una operazione svolta in linea consenta la possibilità di comporre (assiemare) automaticamente materiale nuovo con materiale archiviato e reso disponibile dal sistema; questo in particolare vale per i disegni complessi che ben raramente vengono sviluppati contestualmente alla stesura di un rapporto.
Infine, nella fig. 2 è contemplata anche la possibilità di utilizzare la microfilmatura sia come ingresso e uscita di sistema, che come supporto di memorizzazione; trattandosi di tecnologia già oggi ampiamente utilizzata non si ritiene necessario scendere in ulteriori dettagli.
È chiaro che quando, in una prospettiva oggi ancora di lungo termine, diventerà possibile interagire con l'elaboratore direttamente attraverso la voce, in quel momento tutto il processo di preparazione e trattamento dei documenti qui descritto, dovrà essere sostanzialmente modificato e semplificato; come, è però un compito che va lasciato a chi progetterà non l'ufficio di domani ma quello dell'ancora lontano futuro e che quindi trascende gli scopi di questo articolo.

2.3 - Il sistema integrato

Da quanto si è finora detto risulta logicamente che il sistema informativo per l'ufficio di domani estenderà i principi della elaborazione decentrata a tutti i tipi di comunicazione che fluiscono nella struttura organizzativa unificandoli nella loro rappresentazione digitale e superando così la logica attuale dei sistemi incompatibili e isolati dedicati al trattamento della voce, delle immagini, dei testi e dei dati: da ciò il nome di sistema integrato.
La soluzione tecnologica sarà fornita dalle architetture di informatica distribuita alla cui realizzazione sono attualmente impegnate tutte le principali case costruttrici di elaboratori e in cui sono destinati a confluire gli aspetti oggi ancora critici delle tecnologie di rete, della gestione delle basi di dati decentrate, e del software operativo di supervisione di sistemi complessi con capacità di multielaborazione locale e remota.

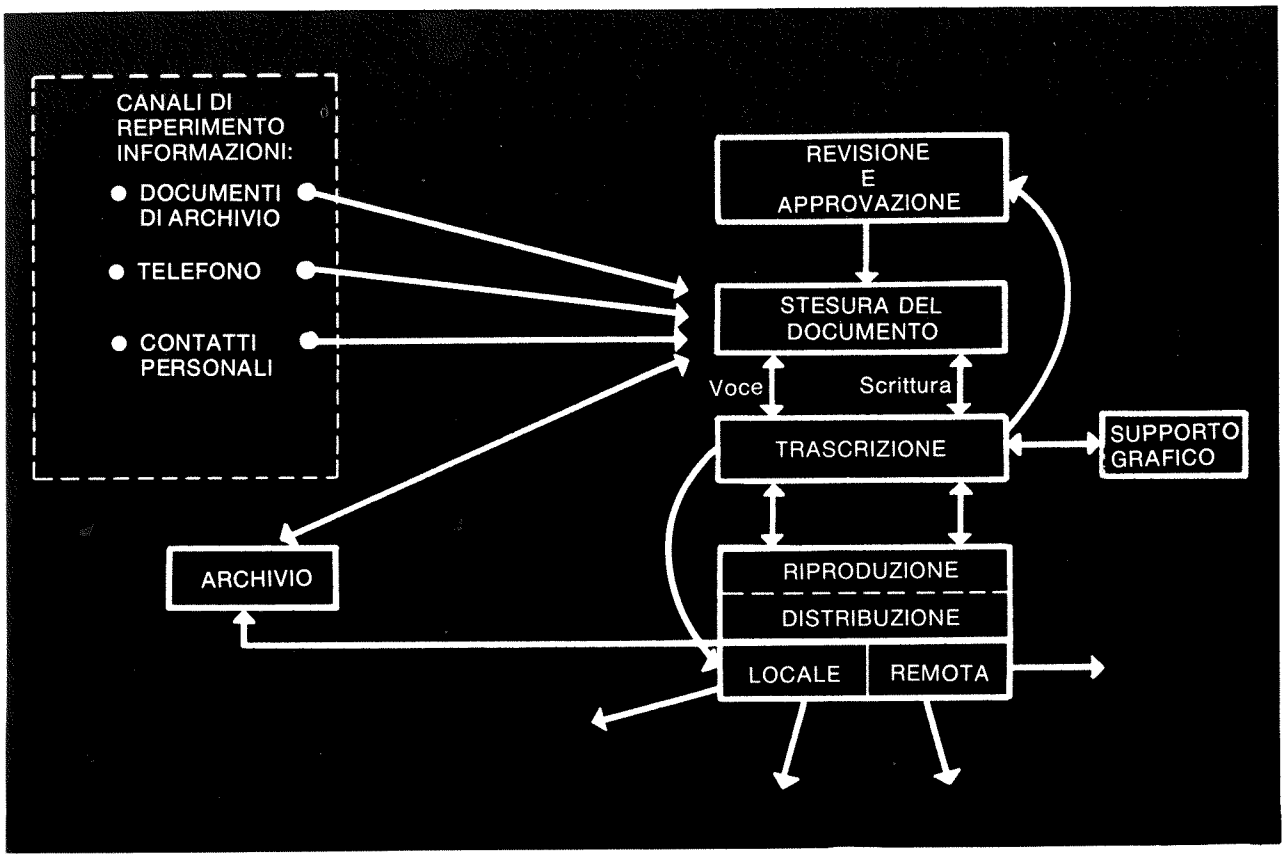


Fig. 4 - Procedura generale di preparazione e distribuzione di un documento

3 - Il lavoro nell'ufficio di domani

Tutto il complesso della attività impiegatizie e direttive subirà nell'ufficio di domani profonde trasformazioni sia quantitative che qualitative rispetto alla situazione di oggi. La tabella 1 presenta un quadro sintetico delle funzioni dell'ufficio e del modo in cui sono destinate a cambiare. In particolare:

— la composizione dei testi/documenti, inclusa la loro eventuale parte grafica, avverrà, come già si è visto, tramite terminale

— la duplicazione dei documenti sarà in larga parte sostituita dalla loro memorizzazione su supporto magnetico e dalla loro visualizzazione su richiesta

— la trasmissione di rapporti, messaggi, promemoria e lettere nonché gran parte delle conversazioni telefoniche, verranno rimpiazzate da comunicazioni asincrome tra terminale e terminale ciascuno dotato di sue proprie «cassette postali» per la corrispondenza in arrivo e in partenza opportunamente protette da accessi indebiti

— le riunioni interpersonali potranno essere in parte sostituite e in parte precedute da «discussioni» a distanza (teleconferencing) fatte tramite terminale riducendo la necessità di spostamenti e forse anche quella di accentramento degli uffici nei grandi complessi urbani

— gli archivi saranno sostituiti da vere e proprie basi di dati e di documenti e la possibilità di accesso e di reperimento in modo strutturato consentirà lo sviluppo di correlazioni più potenti e sintetiche di quanto oggi non consentano la informatica classica e la consultazione degli archivi cartacei

— la gestione dei dati di carattere personale quali gli appunti, la agenda degli impegni, il calendario della attività, i promemoria, etc. potrà essere delegata al sistema in modo tale che le scadenze vengano tempestivamente rammentate agli interessati.

— le attività intellettuali e creative in genere verranno potenziate da queste possibilità nuove di sintesi e di correlazione, cosicché gli stessi processi di valutazione e di decisione tenderanno a mutare.

Nel valutare queste prospettive va ricordato che l'ufficio è il centro di tutto quanto, in una organizzazione, attiene alla generazione e alla diffusione delle informazioni; le applicazioni classiche della elaborazione dati si basano su supporti informativi che hanno la loro origine nell'ufficio; di qui l'interesse della prospettiva di chiudere l'anello, prelevando le informazioni direttamente alla fonte e trattandole poi in modo integrato.

Attraverso l'automazione dell'ufficio, il sistema informativo aziendale acquisisce una dimensione nuova che gli conferisce un ruolo di carattere più specificatamente infrastrutturale.

4 - Aspetti economici

Convinti, come lo siamo, che in un ambito economico, quale quello industriale, i vantaggi ottenibili dalla automazione debbano sempre potersi quantificare per consentirne il confronto con gli investimenti, si pone il problema di individuare i benefici monetizzabili dell'ufficio di domani e le aree in cui sono latenti le maggiori potenzialità.

Settore	Funzione	Equivalente nell'ufficio di domani
Segreteria	Dattiloscrittura	Raccolta dati, elaborazione testi, lettura ottica
	Archivio	Archivi microfilmati, basi di dati, tecniche di accesso e reperimento (Information Retrieval)
	Riproduzione	Stampa
Disegno	Grafici, prospetti, schemi	Elaborazione grafica
Comunicazioni	Interne/esterne: posta, fac-simile, telex, telefono, riunioni formali e informali	EPABX, rete pubblica più linee interne, posta elettronica, teleconferencing
Apparecchiature	Macchine da scrivere e calcolo, dittafoni, fotocopiatrici, telefoni	Terminali multifunzionali, sistemi di stampa (fotocopiatrice intelligente, microfilm, fotocompositrice) e memorizzazione (DBMS estesi ai documenti)

Tabella 1 - Funzioni e servizi nell'ufficio di domani

È ciò che si tenterà di fare in questo paragrafo, in base sia alla analisi delle possibilità che alle risultanze ottenute dagli esperimenti sin qui condotti.

4.1 - Considerazioni generali

L'area applicativa in cui si inserisce la automazione dell'ufficio è quella stessa coperta globalmente dal sistema informativo.
Rispetto a quello proprio dell'informatica classica, è però diverso il tipo di intervento.
Più precisamente, se indichiamo con «elaborazione amministrativa» l'insieme di tutte le attività di ufficio connesse con la utilizzazione delle macchine per scrivere, riprodurre, archiviare, etc., e confrontiamo il costo di tali attività con quelle di elaborazione dati, abbiamo la seguente ripartizione di massima tratta da una valutazione sul mercato USA (1) che però si ritiene sufficientemente indicativa anche per l'Europa:

- costo elaborazione amministrativa 60%
- costo elaborazione dati 40%

Se poi, ulteriormente, confrontiamo queste due voci di spesa con quella da attribuirsi alla più generale voce del lavoro di ufficio, si rileva come, riferendoci sempre alla stessa fonte, la situazione si ponga nei termini seguenti:

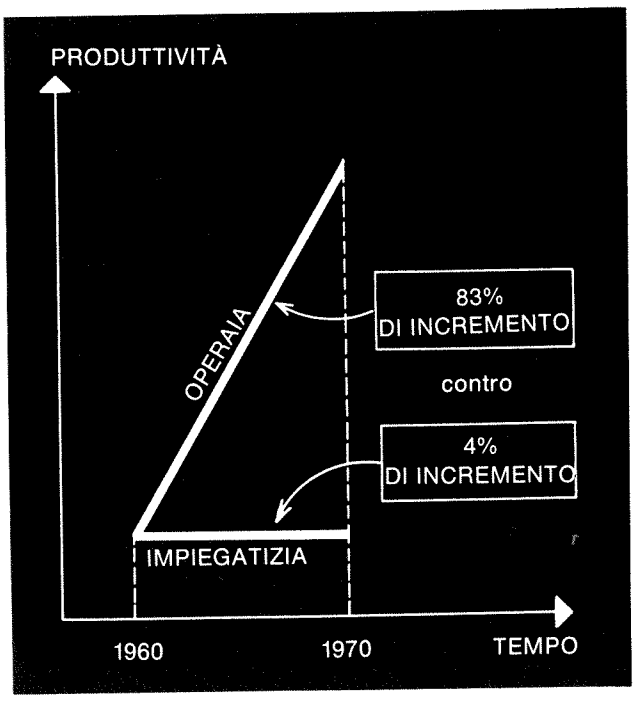
- costo elaborazione amministrativa 10%
- costo elaborazione dati 6%
- costo lavoro di ufficio 84%

È in quest'ultima voce di costo che si propone di intervenire la automazione delle'ufficio.
Se ci riferiamo alle cifre assolute, si tratta di qualcosa che è misurabile in termini di decine di migliaia di miliardi annui per una economia di tipo industriale come quella italiana; si tratta inoltre della componente a più alto tasso di crescita nell'ambito generale del costo del lavoro e con un rendimento per addetto sostanzialmente stazionario se confrontato con quello operaio che nel decennio 1960-1970 ha segnato un incremento superiore all'80% (fig. 5).
L'aumento della produttività operaia è in gran parte il risultato della introduzione di nuovi processi di fabbricazione che hanno comportato un significativo aumento del capitale investito per addetto, capitale che attualmente ammonta a circa 20 milioni di lire; la precedente ripartizione del costo tra personale ed elaborazione amministrativa nel lavoro di ufficio conduce a stimare l'attuale investimento per impiegato come non superiore ai 2 milioni.
Poiché il lavoro impiegatizio consiste sostanzialmente nel trattamento delle informazioni, da queste considerazioni scaturisce la necessità di aumentare significativamente il tasso di automazione dell'ufficio per compensare, con un significativo incremento di produttività, l'ineluttabile aumento dei costi che si verificherà nel futuro.

4.2 - Lavoro segretariale ed esecutivo

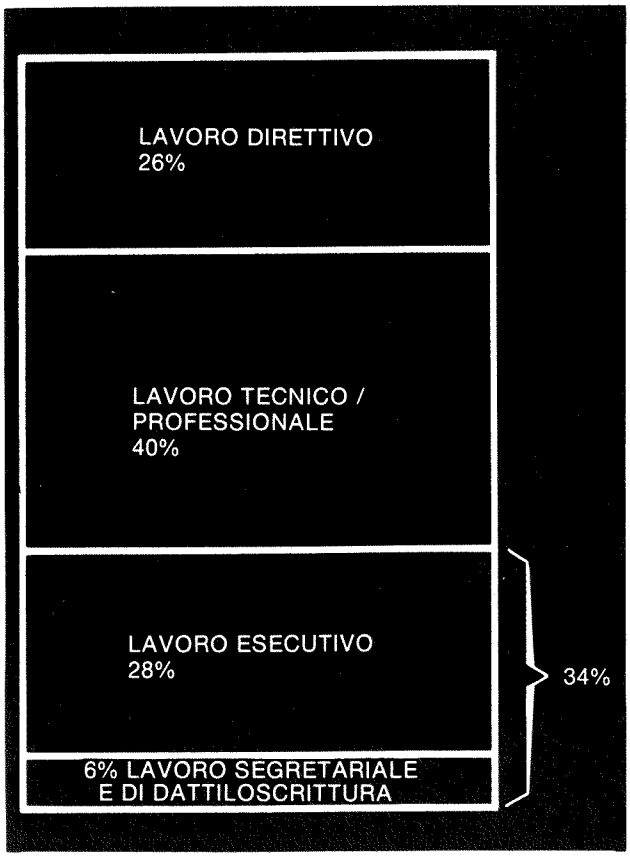
Stando alla dimensione degli investimenti che vi si profondono e alla dinamica tecnologica che la caratterizza, l'area più promettente sembrerebbe essere quella della «elaborazione della parola e dei testi» (word/text processing) e quindi della attività di composizione dei documenti.

Fig. 5 - Andamento della produttività nel decennio 1960-1970



Essendo anche l'area nella quale sono maturate le maggiori esperienze, si è potuto in effetti rilevare che la efficienza nella operazione di dattiloscrittura, può venir migliorata dall'impiego di apparecchiature specializzate, di un fattore che può variare tra 2 e 5; ciò significa che un numero corrispondentemente superiore di righe o parole può venir prodotto per unità di tempo e quindi che il tempo-uomo necessario, a parità di produzione, può essere conseguentemente ridotto.
Il ritorno dell'investimento è, nella fattispecie, misurato dal risparmio che si realizza sul costo del lavoro a parità di produzione dell'addetto alla dattiloscrittura e, dati i costi cui abbiamo accennato al precedente paragrafo, risulta, nella gran parte dei casi, senz'altro significativo.
Tuttavia, se si considera la distribuzione indicativa (2) del costo del lavoro di ufficio nelle sue componenti principali (fig. 6) ci si rende conto che la attività segretariale e di dattiloscrittura incide per una percentuale relativamente modesta (6%) sul totale.
Se poi, ulteriormente, si tenta di scorporare dalla attività segretariale quella propriamente rivolta alla dattiloscrittura (fig. 7), si constata come la incidenza si riduca ulteriormente a non più dell'1,2% (vedi ancora (2)).
Con questo non si vuole evidentemente contestare la utilità degli attuali sforzi nel campo della elaborazione della parola che sono comunque, lo abbiamo già precedentemente affermato, ampiamente giustificati dal punto di vista dell'economia dell'utente, ma solo puntualizzare come non sia questa l'area su cui agire per modificare sostanzialmente la produttività del lavoro di ufficio.

Fig. 6 - Composizione del lavoro di ufficio



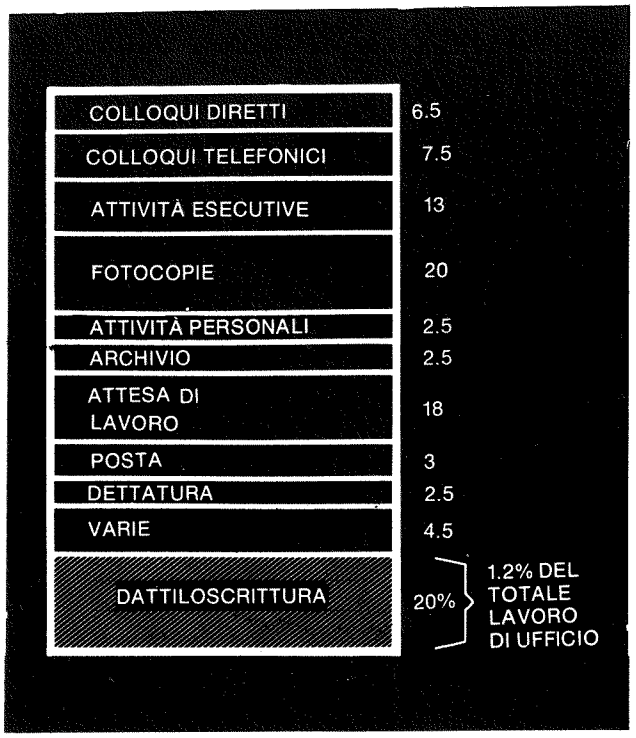


Fig. 7 - Composizione della attività segretariale

Prima di passare all'esame delle altre aree, torna opportuna una considerazione sulla quota, che può apparire sorprendentemente bassa, del costo del lavoro di tipo esecutivo rispetto a quello totale (28%).
Sia pure nei limiti di approssimazione di stime fatte a questo livello di aggregazione, una probabile giustificazione del fenomeno la si può individuare nel fatto che si tratta della fascia di attività in cui si sono andate via via moltiplicando negli ultimi 20 anni le applicazioni dell'elaborazione dati con il risultato che oggi il lavoro esecutivo costituisce sicuramente l'area impiegatizia a maggior tasso di automazione; se questa interpretazione è corretta il fatto costituisce evidentemente un riscontro straordinariamente sintetico e significativo, a livello economico generale, del valore del processo di informatizzazione.

4.3 - Lavoro direttivo e professionale

La struttura del lavoro direttivo è stata e continua ad essere oggetto di indagini approfondite nel mondo economico statunitense; basti qui richiamare le conclusioni dello studio di Mintzberg (3) che sono sintetizzate nella fig. 8.
Da tali conclusioni si deduce che le comunicazioni orali, per telefono o attraverso riunioni, assorbono circa il 75% del tempo lavorativo mentre, delle attività a tavolino, pari al 22%, la gran parte è dedicata allo scrivere e al leggere (20%); il totale del tempo-lavoro dedicato alle comunicazioni essendo quindi del 95%.
Per quanto riguarda il lavoro professionale indagini analoghe (citate in (4), (5) e (6)) evidenziano un 63% totale dedicato alle comunicazioni di cui 37% di tipo orale.
Mediamente, tenendo conto delle percentuali di fig. 6, si può assumere pari all'80% la quota di tempo dedicata alle comunicazioni dai quadri direttivi e professionali della azienda.

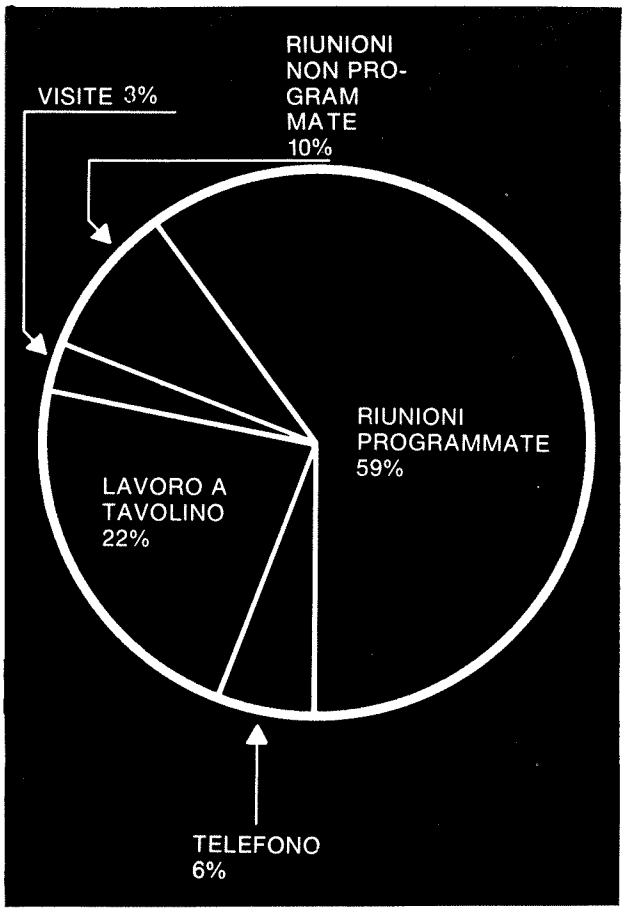


Fig. 8 - Composizione del lavoro direttivo

È a questo punto chiaro che la vera area di intervento capace di modificare sostanzialmente il rendimento del lavoro di ufficio è quella delle comunicazioni; in questa area, è opportuno sottolinearlo, solo recentemente si è inserita l'informatica classica con le applicazioni di tipo interattivo ma è in quest'area che anche essa, convergendo nell'ufficio informatizzato di domani, troverà la sua principale giustificazione economica.

Al punto 3 precedente si è già accennato ai cambiamenti che le nuove tecnologie indurranno nelle attività di comunicazione; si tratta ora di valutare quanto tali cambiamenti apporteranno in termini di riduzione del costo del lavoro o di aumento della sua efficenza.

Anche in questo caso si dovrà purtroppo far riferimento a studi sviluppati in USA in quanto le principali esperienze nel settore derivano dall'impiego di quella che è tutt'ora la più estesa rete di elaborazione e comunicazione esistente nel mondo: la rete ARPA, che collega tra loro i principali centri di ricerca del Nord America.

La rete ARPA, finanziata dal Dipartimento della Difesa USA, non è nata con l'intento esplicito di trattare i problemi dell'ufficio di domani, ma la sua disponibilità ha rappresentato e tuttora rappresenta il migliore strumento di sviluppo e verifica dei progetti relativi a quest'area.

In particolare, la tecnica di trasmissione a commutazione di pacchetto, che è oggi ritenuta la più efficace per questo tipo di applicazioni, è stata sviluppata per la prima volta nella rete ARPA; attraverso la rete ARPA sono stati messi a punto e vengono oggi commercializzati vari package di software dedicati alla gestione delle operazioni di scambio messaggi, ed è sulla rete ARPA che si stanno conducendo i più avanzati esperimenti nel campo delle riunioni a distanza (teleconferencing).

J.M. Bair (2) ha esaminato 4 categorie di benefici che nell'area delle comunicazioni possono tradursi in riduzioni del costo del lavoro, grazie alla posta elettronica, e che derivano dalla:

— minimizzazione delle perdite di tempo che caratterizzano oggi la attività impiegatizia: tentativi più volte ripetuti di stabilire un contatto telefonico, indisponibilità dell'interlocutore al momento in cui lo si cerca, ritardi dei partecipanti ad una riunione, etc.

— eliminazione delle interruzioni nello svolgimento delle attività

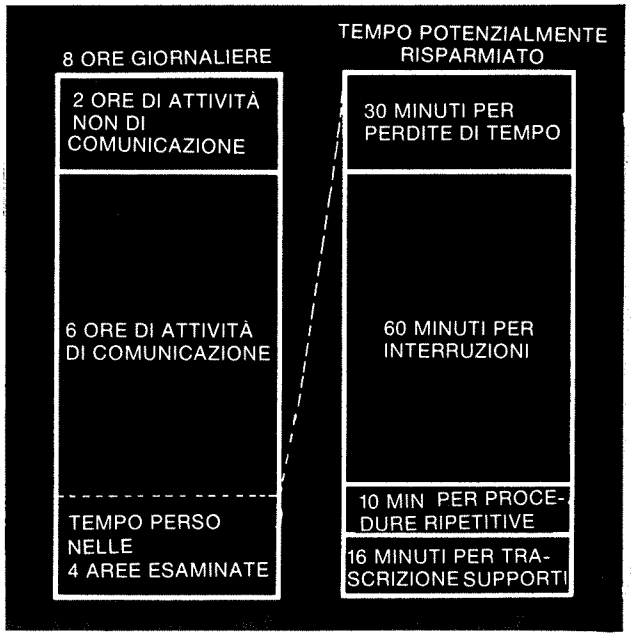
— automazione delle procedure ripetitive di comunicazione per quanto riguarda l'intestazione, la datazione, l'indirizzamento, la distribuzione, la archiviazione, etc.

— normalizzazione dei supporti in quanto nell'ufficio di domani tutte le comunicazioni, con l'eccezione di quelle vocali, che presumibilmente ancora per tutti gli anni 80 continueranno a comportare una trascrizione, verranno gestite dal sistema in modo unificato: la virtuale eliminazione della carta è l'aspetto principale di questo tipo di vantaggio.

Da una analisi accurata dei tempi di svolgimento delle attività direttive e professionali l'autore citato rileva come i benefici indicati possano portare a una riduzione equivalente di circa 2 ore per giorno (fig. 9) con un corrispondente aumento potenziale della produttività.

Se adesso si applicano queste stesse considerazioni a una particolare attività tipica del lavoro professionale di ufficio quale è quella di sviluppo e distribuzione di un rapporto e si fa riferimento allo schema già illustrato in fig. 4, si constata come in questa procedura si possano integrare i vantaggi della posta elettronica sopra accennati con quelli derivanti dalle possibilità di reperire e assiemare le infor-

Fig. 9 - Posta elettronica: risparmio nel lavoro direttivo (e professionale)



Costi relativi / Voci	Ufficio	
	di oggi	di domani
• Personale (tempo)	80	45
• Materiale	13	1
• Elaborazione	—	16
• Distribuzione	7	1
Totale costi diretti	100	63

Tabella 2 - Risparmio relativo nella preparazione e distribuzione di un rapporto

mazioni necessarie direttamente in linea, di comporre il testo con il supporto dell'elaboratore e di distribuire il documento finale per via di trasmissione.
Il risultato, secondo la analisi condotta dal J.M. Nilles (7), è sintetizzato nella tabella 2 dove vengono confrontate le situazioni di costo relativo nell'ufficio tradizionale e in quello di domani.
Come si vede, si può arrivare in applicazioni di questo genere, a riduzioni complessive percentuali che sfiorano il 40% dei costi.
E questo, come sottolinea lo stesso Nilles, con l'impiego di tecnologie già oggi disponibili, valutate al prezzo che hanno oggi sul mercato.
È interessante sottolineare nella tabella 2, la drastica riduzione dei costi di materiale (essenzialmente carta) passando dall'ufficio di oggi a quello di domani e quella, pure notevole, dei costi di distribuzione che nell'ufficio di oggi sono di duplicazione e posta (interna o esterna) mentre in quello di domani sono di comunicazione su linea (interna o esterna).
In questo senso la tabella 2 configura con una certa precisione la dimensione economica dell'ufficio senza carta.

5 - Aspetti organizzativi e sociali

I fattori organizzativi rappresentano in genere un ostacolo al mutamento per la tendenza naturale delle strutture (e delle persone) ad acquisire modalità di lavoro diverse da quelle per cui sono state progettate e secondo cui sono abituate ad operare.
Di fatto la attuale struttura degli uffici con la loro articolazione in centri di costo separati e con i ruoli esecutivi visti come supporto individualizzato a quelli direttivi e professionali, costituisce un'obiettiva difficoltà per la adozione di nuove procedure; la mancanza di una responsabilità unificata per i costi impiegatizi tende d'altra parte a mimetizzarli nella struttura e a rendere anti economici gli interventi necessariamente di portata locale (8).
Una delle principali opposizioni all'ufficio di domani verrà probabilmente dai quadri che oggi si avvalgono di un supporto segretariale personalizzato; già negli anni '60, per le resistenze psicologiche presenti in quest'area, le aziende non hanno potuto attuare quelle modifiche che avrebbero perlomeno razionalizzato la attività di composizione dei testi e che consistevano nel dotare i singoli capi di dittafoni per la generazione del testo e nell'organizzare dei centri intersettoriali di dattiloscrittura per la produzione dei documenti e dei rapporti.
D'altra parte, proprio la dispersione operativa caratteristica del ruolo segretariale e la conseguente scarsa possibilità di crescita professionale che esso oggi offre, è destinata in prospettiva a rappresentare uno dei principali fattori di evoluzione verso l'ufficio di domani: in una società infatti, in cui il lavoro ripetitivo è, in generale, sempre più osteggiato e quindi sempre più difficile a mantenersi, e il movimento femminile tende a conquistare per le donne le stesse opportunità degli uomini, compiti del tipo di quelli schematizzati in fig. 7 è gioco forza che vengano sempre più frequentemente delegati alle macchine.
È questa d'altra parte l'evoluzione già verificatasi in altri settori di attività industriale e tipicamente in quello della produzione, dove la unificazione delle responsabilità e la concentrazione della forza lavoro hanno accelerato il processo di cambiamento con positive conseguenze sia sulla produttività (confronta fig. 5) che sulla stessa qualità del lavoro.
Un altro ostacolo sociale alla adozione delle nuove tecnologie è evidentemente rappresentato dalla riduzione di impiego che potenzialmente determina nella fascia dei servizi amministrativi.
È pur vero che storicamente, dal punto di vista della occupazione, l'informatica presenta un bilancio sostanzialmente positivo, in quanto è stato dimostrato che, nell'arco di un decennio, i posti di lavoro liberati vengono ricostituiti a un livello superiore di professionalità (*); tuttavia questa trasformazione non è stata nel passato priva di crisi e non lo sarà nel futuro specie se dovrà avvenire in un ambito economico senza accentuate prospettive di espansione.
Un'ultima considerazione riguarda le possibilità che l'ufficio di domani prospetta nel campo delle riunioni a distanza (teleconferencing).
La motivazione primaria per l'impiego di sistemi di teleconferencing è evidentemente di carattere economico collegata come è alla possibilità di ridurre i costi comportati dalla partecipazione a riunioni; studi appositi dimostrano che solo gli spostamenti assorbono un 10% circa del tempo di lavoro direttivo o professionale (9).
A differenza della posta elettronica, il teleconferencing comporta che i partecipanti alla sessione siano tutti contemporaneamente presenti sul sistema, in maniera che lo scambio delle comunicazioni avvenga in «tempo reale» per simulare quanto avviene in una vera e propria riunione. I vantaggi consistono nel più diretto confronto tra le

(*) Si ricorda come sino dal 1965 in USA sia stata costituita una specifica commissione per l'esame degli effetti della automazione sulla disoccupazione: la «National Commission on Technology, Automation and Economic Progress», e, come, già nel suo primo rapporto «Technology and the american economy», pubblicato nel 1966 a Washington sia stata esclusa una azione negativa della automazione sull'impiego.
Ad analoghe conclusioni arriva del resto anche P. Stonemann nel suo «Technological diffusion and the computer revolution in the UK economy» Cambridge University Press, 1976, relativamente al caso Gran Bretagna.

opinioni, mentre tra gli svantaggi va citato il tempo limitato che può essere dedicato alla sessione. Con la posta elettronica infatti la «discussione» di un argomento può protrarsi per giorni ed i partecipanti possono approfondirne con cura i vari aspetti.
Senza voler entrare nel merito delle diverse impostazioni tecnologiche che si contendono il mercato basti dire che alcune grandi aziende statunitensi hanno già installato con soddisfazione sistemi di questo genere per collegare tra loro sedi remote e che, nella prospettiva di automazione dell'ufficio, intendono avvalersene come di un importante complemento alla posta elettronica.
Si pone quindi il problema di valutarne l'impatto in termini organizzativi e di comunicazione.
L'esperienza disponibile non è ancora sufficiente per dare una risposta completa, tuttavia le prime risultanze sembrano confermare che il teleconferencing è uno strumento che integra soddisfacentemente i servizi forniti dalla posta elettronica e che consente da solo di arrivare ad una conclusione quando l'argomento sia di carattere prevalentemente tecnico; nel caso, invece in cui l'aspetto politico sia predominante, una vera riunione risulta comunque opportuna ma dopo che con la posta elettronica e con una o più sessioni di teleconferencing, i partecipanti siano stati tutti portati allo stesso livello di informazione e di comprensione dell'argomento.

6 - Stato applicativo e tendenze di ricerca

Si è visto come l'ufficio di domani sia tecnologicamente già una realtà, ma sistemisticamente e organizzativamente sia ancora un argomento di ricerca.
Si ritiene quindi non inutile accennare brevemente gli attuali orientamenti applicativi e alle principali direttrici di ricerca e sviluppo.
Per quanto riguarda il primo punto le organizzazioni industriali che stanno oggi affrontando con più decisione i problemi della automazione dell'ufficio appartengono in generale, al settore finanziario: grandi Istituti di Credito o Compagnie di Assicurazione.
Da una parte perché si tratta della utenza con maggiore esperienza nell'impiego dell'informatica e dall'altra perché si tratta di aziende in cui le attività di trattamento delle informazioni risultano assolutamente predominanti.
L'area principale di intervento è evidentemente quella delle attività ripetitive ma in breve tempo, nell'attuare il processo di integrazione tra la informatica classica e quella di ufficio, si arriva a coinvolgere le attività di carattere più specificatamente direzionale dove, come già si è osservato, maggiore è il potenziale di ritorno e dove, in questo tipo di aziende, già è consolidato l'impiego di basi di dati ad alto livello di integrazione.
Un altro settore in cui l'automazione dell'ufficio, con particolare enfasi sugli aspetti di comunicazione, è da anni una specifica realtà applicativa, è quello delle varie organizzazioni militari statunitensi; anzi, si può affermare che, come tanta parte dello sviluppo dell'informatica, anche l'ufficio di domani sia sostanzialmente una derivazione degli investimenti nella ricerca militare; e basti qui ricordare la già citata rete ARPA (Advanced Research Project Agency) che, pur rappresentando oggi un potente strumento di ricerca non finalizzato a scopi bellici, è tuttavia stato finanziato dal Dipartimento della Difesa.
Per l'esame di casi specifici di applicazione si rinvia agli articoli comparsi con una certa frequenza negli ultimi 18 mesi e che in parte vengono richiamati nella bibliografia.
Sul secondo punto, invece, si elencano nel seguito alcune attività in corso che complementano quelle già precedentemente accennate relative alla rete ARPA.

Telenet

Si tratta di una rete di servizio, come si dice in inglese VAN per Value Added Network, a commutazione di pacchetto, che ha mutuato dalla ARPANET sia la tecnologia che la struttura.
In un sistema a commutazione di pacchetto (10) i messaggi vengono suddivisi in «pacchetti» (di 128 caratteri nel caso telenet) e trasmessi in parallelo da un nodo all'altro su quei canali che, di volta in volta, si presentano liberi.
Il sistema non richiede linee private tra i punti terminali e si sta dimostrando molto più conveniente e affidabile di tutte le altre alternative di comunicazione. Entro il 1980, ad esempio, si prevede che un tipico messaggio (sino a 1000 caratteri di lunghezza) potrà essere trasmesso sulla rete Telenet da qualsiasi punto a qualsiasi altro nel mondo con una spesa variabile tra 0.1 e 0.3 dollari.
La rete Telenet è gestita da una organizzazione privata e, dagli Stati Uniti, si sta gradualmente estendendo all'Europa.

Bell Canada

Il Business Planning Group di questa organizzazione è da anni impegnato in ricerche sul «Computer teleconferencing», cioè sulla possibilità teorica e pratica di organizzare riunioni a distanza con supporto di elaboratori come alternativa al trasporto. Alcuni esperimenti si estendono anche all'area della automazione della casa. Va ricordato che queste ricerche si inquadrano nel progetto più generale, varato dal governo federale canadese nel 1969, per lo studio delle implicazioni industriali economiche e sociali della informatica e delle comunicazioni (la cosiddetta tele-informatica) nello sviluppo del paese.
Il documento di riferimento per tutto il progetto è stato pubblicato nel 1972 con il titolo «l'Arbre de vie» successivamente rielaborato e completato nel «Livre vert» del 1973.

Communications Studies Group, Joint Unit for Planning Research, University College (London)

Si tratta di uno dei gruppi di ricerca più autorevoli per quanto riguarda gli aspetti psicologici dell'ufficio automatizzato e la valutazione dell'impatto delle comunicazioni sulle varie funzioni.
Il gruppo è parzialmente finanziato dal British Post Office.

New Jersey Institute of Technology

Il Computer and information Science Department ha sviluppato e reso operativo sin dal 1977 con il supporto della National Science Fondation, un sistema sperimentale di teleconferencing per scopi di ricerca. I componenti principali del sistema sono:

- *Notebook:* una sorta di agenda di lavoro personale per la preparazione di bozze di messaggi, commenti e rapporti
- *Message:* un sistema di comunicazione per lo scambio di messaggi nell'ambito di un gruppo di lavoro
- *Conference:* un sistema per «riunioni» intergruppi che possono tenersi sia in modo sincrono (tutti i partecipanti contemporaneamente presenti al loro terminale) oppure asincrono; in ambedue i casi il sistema provvede alla stesura del «verbale» della riunione;
- *Bulletin:* per informare periodicamente i vari gruppi di lavoro di fatti nuovi o comunicazioni brevi di utilità generale.

Stanford Research Institute

È già stato citato più volte nel corso dell'articolo in quanto rappresenta uno dei poli di sviluppo, a livello mondiale, delle ricerche sulla informatica in generale e sulla automazione dell'ufficio in particolare.
Un gruppo di lavoro, detto ARC per Augumentation Research Center, si occupa dal 1964 dello sviluppo di tutta una serie di strumenti software volti a migliorare e potenziare il rapporto di interazione uomo-macchina; il progetto prende complessivamente il nome di NLS per oN Line System.
Parti di questo sistema sono di uso corrente nella rete ARPA.

University of Pennsylvania

La Wharton School of Management ha sviluppato un progetto integrato di automazione di ufficio e di supporto decisionale.
Il sistema è costituito da due moduli principali:

- *Oasys* (Office Automation System) che è sostanzialmente un supervisore di coordinamento dei vari sottisistemi funzionali relativi alla automazione dell'ufficio e al supporto direzionale
- *Daisy* (Decision Aiding Information System) che ha lo scopo di fornire ai responsabili una base integrata di dati testi e grafici, capace di sopportarli interattivamente nel processo decisionale, ed è in parte operativo ed in parte in fase di sperimentazione

NASA (National Aeronautics and Space Administration)

Dal 1974 si occupa sistematicamente di applicazioni di teleconferencing nell'ambito di tutta la sua organizzazione, ma già nel 1969-70 aveva ottenuto notevoli successi con i sistemi sperimentali adottati durante lo sviluppo del programma APOLLO. Per dare un esempio della efficacia di impiego di questi sistemi in ambito NASA, basti citare il fatto che nel 1975, grazie ad essi, si sarebbero risparmiati circa 4600 viaggi pari a 1.28 Milioni di dollari (il 10% circa dell'intero budget costi di trasporto della NASA).

University of Southern California

È un altro centro di grande importanza nel campo generale della automazione dello ufficio e si è già avuta occasione di citarlo per il contributo di J.M. Nilles; opera su più fronti:

- *Tecniche di analisi*

Lo scopo è di fornire metodologie di analisi del flusso informativo aziendale che possano rappresentare un riferimento oggettivo nello sviluppo di progetti; si tratta di tecniche concettualmente simili a quella discussa in (11).

- *Analisi costi/benefici*

Studio di alternative organizzative ed economiche in diversi tipi di scenari operativi con particolare interesse agli aspetti di decentramento e di sostituzione delle comunicazioni al trasporto.

- *Servizi di rete*

Tratta in generale gli aspetti di teleinformatica di cui l'ufficio di domani è una conseguenza.

- *Software per la automazione dell'ufficio*

Si avvale della rete ARPA per lo sviluppo di prodotti orientati a facilitare l'impiego dell'informatica da parte di personale non specializzato.

- *Analisi delle funzioni di ufficio*

Indaga la attuale struttura degli uffici e, in funzione dei compiti che vi si svolgono, propone metodologie di razionalizzazione del flusso informativo: è il presupposto della attività di analisi costo/benefici.

7 - Conclusione

La storia dell'informatica è ricca di mode fondate su presupposti tecnologici inconsistenti e assunzioni rivelatesi poi del tutto ingiustificate.
Basti qui ricordare la mitologia del «Sistema Totale» che tanto peso ha avuto negli anni '60 e che si proponeva di trasformare la azienda in un sistema cibernetico controllabile con l'elaboratore nella evoluzione di tutti i suoi principali gradi di libertà; l'unico spazio che in questa utopia veniva lasciato all'uomo era quello di definire gli obiettivi generali e di intervenire in presenza di eventi del tutto imprevedibili.
Il fatto più sorprendente è che ci sia stato chi ha preso sul serio una simile possibilità e ha profuso energie e investimenti nel tentativo di realizzarla in concreto; i risultati, non è neanche il caso di ricordarlo, sono stati disastrosamente fallimentari, tanto di assumere un valore paradigmatico.
In questo articolo si è cercato di dimostrare che l'ufficio di domani non ha niente a che fare con questo tipo di mitologie: si tratta di qualcosa di sostanzialmente diverso la cui concezione poggia su solide considerazioni economico/organizzative e il cui investimento in ricerca e sviluppo è tale da garantire nel breve medio termine una sua applicabilità estesa al mondo produttivo con precisi riscontri di efficienza.
Alcune previsioni valutano che nel 1985 metà del fatturato dell'industria informatica deriverà dalla automazione dell'ufficio.

8 - Bibliografia

(1) The Diebold Research Program Europe, Conference Proceedings Meeting XXXIV, Madrid, 1975.
(2) J.M. Bair, *Communication in the office of the future,* Stanford Research Institute International, 1978.
(3) H. Mintzberg, *The nature of managerial work,* Harper and Row, New York, 1973.
(4) R.R. Panko, *Presentation to the seminar on the augumented knowledge workshop,* Stanford Research Institute, 1976.
(5) J.M. Bair and D.W. Courath, *The computer as an interpersonal communication device: a study of augmentation technology and its apparent impact on organizational communication,* Proccedings of the 2nd International Conference on Computer Communication, Stockholm, 1974.
(6) G.R. Edwards - *Analysis of usage and related perceptions of NLS - A computer based text processing system,* Business development, Bell Canada, Montreal, 1977.
(7) J.M. Nilles, *Strategies in moving towards the office of the future,* The Diebold Research Program Europe, Conference Proceedings Meeting XXXIX, London, 1977.
(8) A. Purchase, *Office of the future,* Stanford Research Institute, 1976.
(9) R.R. Panko, R. Pye and R.W. Hough, *Telecommunications for the office decentralization: apparent needs and investment requirements,* Proceedings IEEE Int. Conference on Communication, Philadelphia, 1976.
(10) D.A. Walsh - *Reti di eleboratori: la trasmissione dei dati con commutazione di pacchetto* - Quaderni di Informatica N. 9, 1978.
(11) P. Blasio - *La progettazione dei sistemi informativi aziendali.* Quaderni di Informatica N. 7, 1977.

Alcuni articoli o rapporti su applicazioni di automazione ufficio

J.P. De Blasis - *Office automation systems: another possible route to MIS,* Proccedings IEEE Int. Symposium on technology for selective dissemination of information, S. Marino, 1976.
J.C. Gilbert, *Computer based office support system: electronic mail,* - The Diebold Research Program Europe, Conference Proccedings Meeting XXXIX London, 1977.
R.B. White, *A prototype of automated office,* Datamation, April 1977.
J.C. Burns, *The evolution of office information systems,* Datamation, April 1977.
C.B. Carls. *Getting ready for word processing's second generation,* Datamation, September 1978.
H. Anderson, *Consider the Bank of America,* Datamation, September 1978.

NOTE

Questo articolo è tratto da un libro dell'Autore, di prossima pubblicazione nella collana dei «Quaderni di Informatica» dell'editore F. Angeli.

Animazione grafica mediante «computer output microfilm»

GIUSEPPE FENINI
Enel
Direzione Studi e Ricerche
Centro Ricerca Idraulica e Strutturale
Milano

1 - Introduzione

L'interesse suscitato dalle tecniche di graficizzazione automatica, dei risultati numerici ottenuti da elaboratori elettronici, è andato sempre più crescendo negli ultimi anni a questa parte.
La sintesi che la rappresentazione grafica permette di operare sulla marea di informazioni in uscita dall'elaboratore, è un prezioso ed insostituibile aiuto per una rapida valutazione tanto qualitativa che quantitativa degli elaborati ottenuti.
Questo vale in modo particolare quando ci si riferisce all'utilizzo di elaboratori per la soluzione di complessi problemi di tipo scientifico: un codice di calcolo in grado di fornire uscite grafiche automatiche trova maggior dipsosizione al suo utilizzo di quelli, anche più sofisticati, che si limitano ad esibire alla fine una pura e spesso prolissa esposizione numerica.
Il film di animazione, ottenuto automaticamente dalle informazioni generate dall'elaboratore, può essere a buon diritto considerato l'attuale frontiera dell'espressione grafica automatica. Con l'aggiunta della dimensione temporale, manipolata a piacere come tutte le atre cui il grafi̇cismo già accadeva, consente di percepire la continuità di quei fenomeni che evolvono nel tempo e per i quali altrimenti non è possibile fornire che dei flasches discontinui.
Una animazione consiste sostanzialmente di una serie di fotogrammi proiettabili su schermo con gli usuali apparati per proiezione cinematografica. Ogni fotogramma continee l'informazione grafica relativa ad un certo istante dell'evoluzione di un fenomeno (descritto da un codice di calcolo operante su elaboratore); la proiezione sequenziale di fotogrammi restituisce quindi allo spettatore lo svolgimento di questo fenomeno in una dimensione temporale la cui ampiezza dipende dal rapporto che intercorre fra la durata delle scansioni di generazione dei fotogrammi e quella della loro proiezione.
La realizzazione di una animazione si basa su di una catena di elementi di natura software ed hardware che si possono così sintetizzare:

a) programmi che interpretano i fenomeni che si vogliono rappresentare;
b) programmi per la visualizzazione grafica delle diverse situazioni temporali;
c) routines di generazione dei dati (HCBS = Host Computer Basic Software) necessari per il funzionamento dell'unità che impressiona il film (COM = Computer Output Microfilm);
d) elaboratore ove operano gli elementi precedenti;
e) interfaccia elaboratore-COM grafico;
f) COM grafico (unità che fisicamente genera il film) e pellicole cinematografiche;
g) sviluppatore del film;
h) proiettore.

2 - Il Software

È raggruppato nelle prime tre voci esposte, con la precisazione che normalmente le routines di generazione dei dati utili al COM grafico (HCBS) vengono fornite dallo stesso costruttore del COM grafico ed utilizzate dagli altri programmi con opportune chiamate.
Si hanno le condizioni operative più semplici quando il codice di calcolo descrivente il fenomeno da visualizzare comprende anche il programma grafico relativo e le routines di generazione dati. La semplice esecuzione del codice genera il flusso dei dati necessari alla realizzazione grafica. Questo è ad esempio il caso di procedure esistenti ed assemblate con routines HCBS di generazione dati per «plotter» (bisualizzatore grafico equipaggiato con penne scriventi, normalmente su fogli o rotoli di carta). Gli stessi dati che vengono utilizzati da un plotter sono adatti, nella maggior parte dei casi con qualche semplice variante nelle HCBS, all'ultimizzo su di un COM grafico.
Si verifica invece spesso la necessità di suddividere in esecuzioni diverse le elaborazioni relative ai calcoli veri e propri e far seguire a queste, in tempi differiti, la generazione dati per la visualizzazione grafica. Questa condizione operativa più complessa è realizzabile operando un'accorta gestione dei dati in transito fra le varie fasi

previste e consente di raggiungere due importanti risultati: spezzare quelle esecuzioni che risulterebbero altri menti particolarmente lunghe e utilizzare proficuamente le possibilità della multiprogrammazione.
Occore evidenziare il fatto che la rapidità di esecuzione dei programmi divisualizzazione è per questa tecnica un requisito fondamentale. Ogni minuto di proiezione, alla velocità di 16 fotogrammi/sec., richiede circa 1000 fotogrammi; perciò ogni accorgimento atto ad accelerare anche di poco i tempi di esecuzione, assume qui grande importanza.
Fortunatamente, l'uilizzo del COM grafico non richiede come per il plotter l'ottimizzazione dei percorsi nel tracciamento dei grafici: non esiste la necessità di minimizzare gli spostamenti dell'equipaggio scrivente a penne alzate, problema la cui soluzione i programmatori di codici grafici sanno quanto spesso sia causa di complessità e quindi di aumento dei tempi di esecuzione.

3 - L'elaboratore

In linea di principio non vi sono restrizioni all'uso di un elaboratore rispetto ad un altro. Va tenuto presente che l'elevato numero di disegni richiesti provoca un intervento massiccio dei programmi e delle routines di disegno, quindi la potenzialità di elaborazione disponibile è opportuno si collochi in una fascia medio-alta.
Spesso vengono utilizzate le risorse di grandi e potenti elaboratori remoti da un piccolo terminale locale collegato con una linea telefonica dedicata, sulla quale le informazioni transitano ad una velocità che nel migliore dei casi arriva a 9600 Baud. In queste condizioni è del tutto sconsigliabile inviare sulla linea i dati necessari al COM grafico: nel caso di una animazione la loro quantità è sempre talmente elevata che paralizzerebbe l'attività del terminale ricevente dilatando i tempi di esecuzione in modo inacettabile.
Dovendo perciò ricorrere ad un utilizzo remoto dell'elaboratore è preferibile archiviare, al momento della loro generazione, i dati necessari al funzionamento del COM grafico su di un supporto fisico compatto e facilmente trasportabile (ad esempio nastro magnetico) ed utilizzarli successivamente in locale evitando così di impegnare la linea trasmissione dati.

4 - L'interfaccia elaboratore - COM grafico

Si intende qui per interfaccia l'insieme di tutti quei dispositivi hardware e software necessari al trasferimento ed all'interpretazione dei dati generati dalle HCBS in comandi operativi per il COM grafico.
Le routines HCBS generano dei dati in una forma il più sintetica e compatta possibile per ridurre al minimo il carico di elaborazione sul computer che le ospita e l'impegno del canale di trasferimento dei dati stessi al supporto utilizzatore. Nasce quindi l'esigenza di un dispositivo adatto a recepire queste informazioni ed a trasformarle in tutti quei comandi elementari necessari al funzionamento del COM grafico. Questo dispositivo, che prende il nome di CONTROLLER, è un'unità che può essere interfacciata all'elaboratore principale, in linea o in modo differito (figg. 1 e 2).
Nel primo caso, da un canale imput/output perverranno al controller i macro dati generati direttamente dalle HCBS (o da uno spooler che li riprenderà da un supporto di parcheggio al termine di una esecuzione); nel secondo caso, generato con le HCBS sull'elaboratore principale un flusso di macro dati e depositatolo su un conveniente supporto, questo verrà letto da un'adatta unità del controller stesso.
Si tratta quindi di un dispositivo intelligente, capace cioè di accettare informazioni seriali scandite in modo casuale e di elaborarle esplodendole in microinformazioni nei tempi intercorrenti fra un invio e l'altro. È necessario perciò che possa attendere fra invii successivi o temporizzare l'accettazione qualora l'elaborazione risultasse più lunga dei tempi di interrecord-gap.
Normalmente l'elaboratore che equipaggio il controller è istruito da un programma specializzato nel regolare il traffico delle informazioni in ingresso, cioè provenienti dall'elaboratore principale o da supporto magnetico, e, dopo opportuna interpretazione ed elaborazione, quelle in uscita destinate al COM grafico.
I modelli più completi sono in grado di pilotare, utilizzando canali di 1/O diversi, tanto i plotters quanto il COM grafico, anche se non contemporaneamente.
Le stesse case costruttrici dell'hardware preparano diversi tipi di programmi ognuno dei quali è adatto alle caratteristiche ed al modo di acquisizione dei macro-dati in ingresso secondo il tipo di elaboratore principale usato.

Fig. 1 - Operazione on-line.

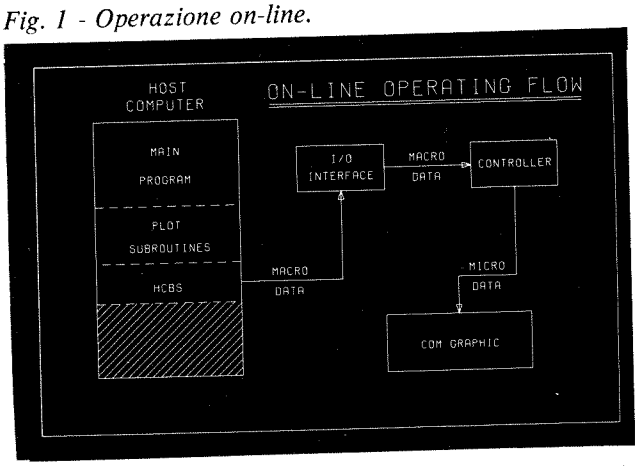
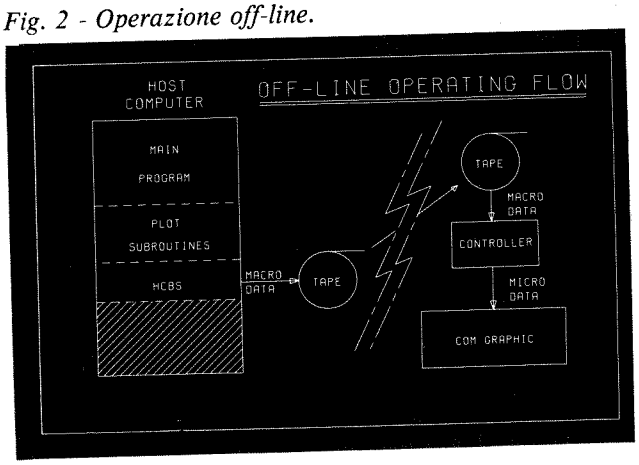


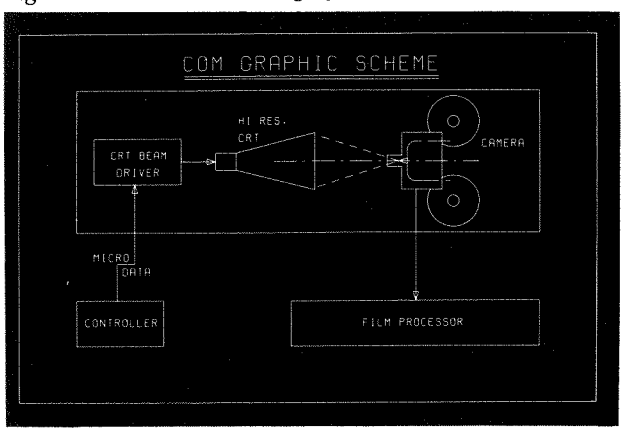
Fig. 2 - Operazione off-line.



5 - Il COM grafico

È questo uno degli elementi più importanti di tutta la catena, ed ha il compito di traudrre in immagini grafiche i dati passati al controller. (fig. 3).
Caratteristica fondamentale del COM grafico è quella di restituire su di uno schermo perfettamente piano immagini di elevata nitidezza ed equale omogeneità in tutti i punti dello schermo stesso. Questo è ottenuto con un tubo catodico a schermo piano antiriflettente, sul fosforo del quale interviene un pennello elettronico monocromatico pilotato da circuiti tali da assicurare la massima linearità di tracciamento.
Il numero dei punti distinguibili sullo schermo forma un reticolo che vale 16384 x 16384 unità, formando quindi una matrice quadrata nella quale si hanno quasi oltre 270 milioni di posizioni distinte. Questo significa, in fase di proiezione su di uno schermo di m. 1,50 x 1,50, disporre potenzialmente di una risoluzione del decimo di millimetro.
Il pennello elettronico può assumere una trentina di diverse intensità luminose selezionate automaticamente dalle informazioni fornite dalle routines HCBS o, sovrapponendosi a queste, da un'opportuna facile variazione della programmazione del controller. L'immagine prodotta è «vista» da una camera a lenti fissata di fronte allo schermo sul suo asse longitudinale.
La camera, che può essere alimentata da diversi formati di pellicola sensibile, non è provvista di otturatore ed ogni fotogramma viene impressionato dai successivi movimenti del pennello elettronico, la cui traccia non ha nessuna persistenza per evitare variazioni di esposizione fra le prime e le ultime linee di uno stesso disegno. L'avanzamento dei fotogrammi è ottenuto automaticamente ogni qualvolta si completa un disegno: anche questa è un'informazione generata dalle routines HCBS o direttamente impostabile sul controller.
Un'ulteriore importante caratteristica del COM grafico è la elevata rapidità di esecuzione del prodotto disegno: la velocità di tracciamento è tale da consentire il disegno di 20 ÷ 80 linee al secondo lunghe quanto tutta l'ampiezza del campo visibile: ciò consente di realizzare animazioni anche di diverse migliaia di fotogrammi in tempi compresi nell'arco di una giornata.
Le pellicole cinematografiche impiegate debbono avere elevatissime caratteristiche di risoluzione al fine di ottenere un utilizzo massimo della precisione dello schermo visualizzante. Si tratta di prodotti sensibili speciali fabbricati per questi tipici impieghi; hanno bassa sensibilità, elevato contrasto e richiedono prodotti chimici non usuali per il loro trattamento.
Il formato utilizzabile è vario perché tanto le camere di ripresa quanto le lenti sono intercambiabili. Sarà il tipo di utilizzo finale a suggerire la combinazione più conveniente tenuto conto anche del tipo di proiettore impiegato.

Fig. 3 - Schema di COM grafico.



6 - Lo sviluppatore

Questo dispositivo permette di ottenere: sviluppo, processo di inversione, fissaggio, lavaggio, asciugatura e bobinatura della pellicola impressionata dal COM grafico in modo completamente automatico e non in ambiente oscurato.
Non è pensabile la realizzazione di una animazione senza affiancare al COM questo apparato, in quanto è molto importante poter visionare immediatamente tutte quelle prove che l'utilizzatore esegue, modificando i diversi parametri di funzionamento del COM stesso, durante la fase iniziale di messa a punto delle immagine grafica.

7 - Il proiettore

Per la visione dell'animazione è sufficiente un proiettore cinematografico dalle caratteristiche standard. La scelta del passo dipende esclusivamente dalla qualità richiesta per le immagini proiettate, ma a questo proposito va detto che il formato 16 non consente di ottenere risultati qualitativamente molto validi senza dover impegnare l'utilizzatore nella gestione di proiettori di non facile impiego quali sono quelli per i formati maggiori.

8 - Applicazioni

I fenomeni fisici che per mezzo di una rappresentazione animata rivelano al ricercatore aspetti interessanti o addirittura determinati ai fini della ricerca, sono moltissimi e costituiscono tutto un mondo da scoprire.
Alcune animazioni sono state realizzate nel nostro Centro di Ricerca, con buona riuscita ed incoragginati consensi.
Un primo esempio tratta di un transitorio termico allo scarico dell'acqua di raffreddamento dei condensatori di una centrale termoelettrica situata sulle sponde di un fiume.
L'approccio al problema, che riguarda la diffusione termica in acque fluviali, è affrontato con modello fisico per la generazione dei dati di base e con modello matematico per la visualizzazione grafica.
Il modello fisico rappresenta, con opportune scale, un tratto di fiume prossimo alla foce marina; la portata è variabile ciclicamente nelle 24 ore del giorno per effetto della marea; la diffusione è anch'essa variabile a causa del cuneo salino montante dalla foce. Nello stesso tratto, che presenta le principali diramazioni del delta, si trovano i canali di presa e restituzione delle acque per il raffreddamento dei condensatori di una ipotetica centrale termoelettrica.

Giunti a regimazione idrotermica, vengono rilevate le temperature superficiali in alcuni punti dell'area liquida con opportune scansioni temporali tali da ricoprire, tenuto conto della distorsione della scala dei tempi, l'intero ciclo di marea. Questi dati vengono forniti in ingresso ad un programma di calcolo la cui maglia di discretizzazione ed elementi finiti ricopre l'intera superficie liquida: alcuni vertici degli elementi discretizzanti coincidono geometricamente con i punti di rilevamento delle temperature. Per ogni situazione termica misurata, il programma, utilizzando l'equazione di Laplace per campi bibimensionali, è in grado di assegnare ad ogni punto del continuo un valore interpolato della funzione rappresentata, che in questo caso è la temperatura superficiale.
Poiché il rilevamento termico non può essere fatto con scansioni tanto rapide quanto sarebbe richiesto dall'animazione, vengono determinate le situazioni intermedie per ogni nodo della maglia con una interpolazione polinominale del terzo ordine fra le scansioni effettuate (utilizzando i polinomi di Lagrange). Questa procedura consente, con un notevole risprmio dei tempi di elaborazione, di ottenere dei flasches ogni centesimo di ora, partendo da situazioni rilevate con scansioni di durata due ore ciascuna.
Ogni situazione così generata viene quindi esaminata da un programma per il tracciamento automatico di linee isoterme che costituiscono l'immagine sul fotogramma.
Per questa animazione si hanno allora 24/0,01 = 2400 disegni diversi che, ripetuti ciascuno due volte, danno luogo a 4800 fotogrami proiettabili in 5 minuti primi alla vleocità di 16 fotogrammi/s. Si tratta quindi di una rappresentazione accelerata 288 volte nel tempo.
La visione dell'animazione ha permesso di mettere in evidenza alcuni aspetti caratteristici del fenomeno osservato, come: l'inerzia termica del sistema, l'entità dello sfasamento fra azione della marea e risposta del flusso termico, la variazione di forma e addensamento delle isoterme, la durata del fenomeno diffusivo nell'aria, ed ancora altri interessanti aspetti.
Va da se che la valutazione precisa di ognuna di queste condizioni è alla fine da ricercarsi nei valori numerici esiviti dalle elaborazioni, ma è molto importante poter «aggiustare il tiro» nella ricerca di quei pochi dati significativi fra la moltitudine di quelli generati ed avere una visione d'insieme del fenomeno con la contemporaneità e la sovrapposizione dei diversi elementi in gioco.
Le figure 4 - 15 rappresentano situazioni temporali distanti 2 ore una dall'altra.
Una seconda animazione è stata realizzata traendo lo spunto dalla necessità di verificare visivamente come poteva collocarsi la presenza di un edificio da erigere fra alcuni altri esistenti.
I volumi da rappresentare sono stati discretizzati con elementi tridimensionali isoparametrici (vale a dire descritti nello spazio da piani e linee curve del secondo ordine) posti in uno spazio cartesiano ortogonale.
Un programma specializzato consente di tracciare trasformazioni prospettiche del continuo così discretizzato,

Figg. 4-15 - Animazione del transitorio termico allo scarico dell'acqua di raffreddamento dei condensatori di una centrale termoelettrica situata sulle sponde di un fiume. Le figure rappresentano situazioni temporali distanti 2 ore una dall'altra.

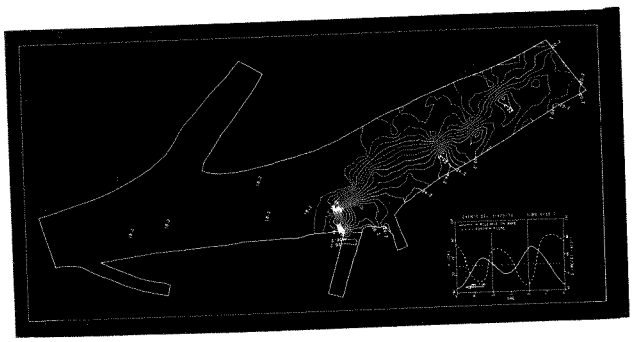


Fig. 4

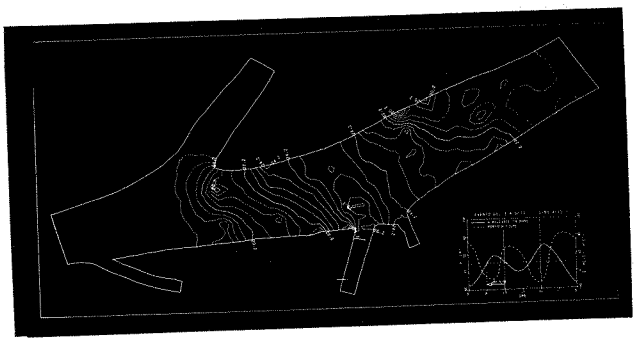


Fig. 6

Fig. 5

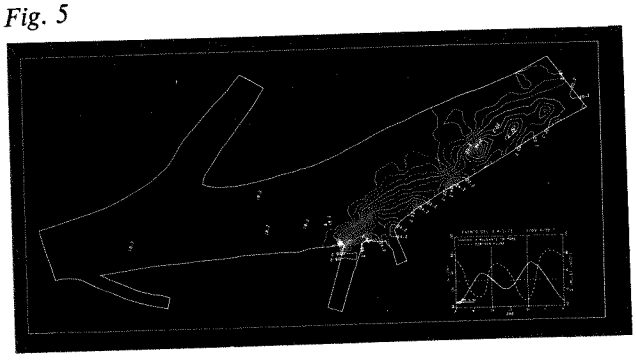
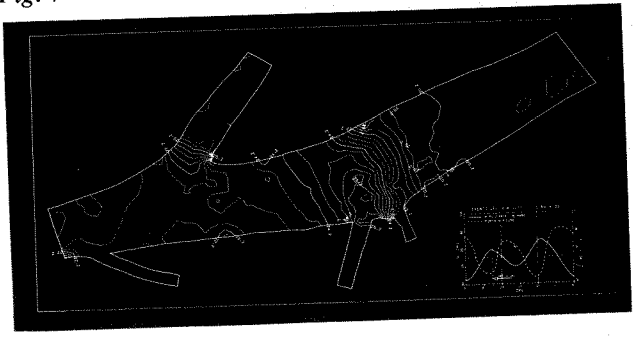


Fig. 7



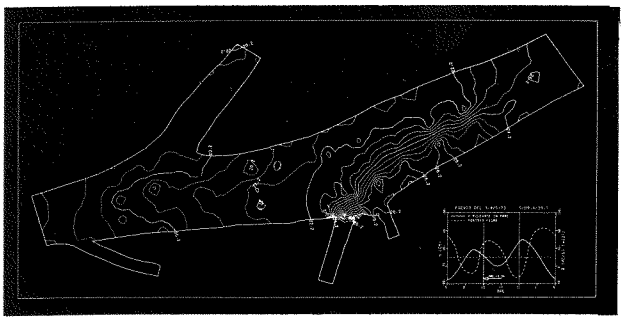


Fig. 8

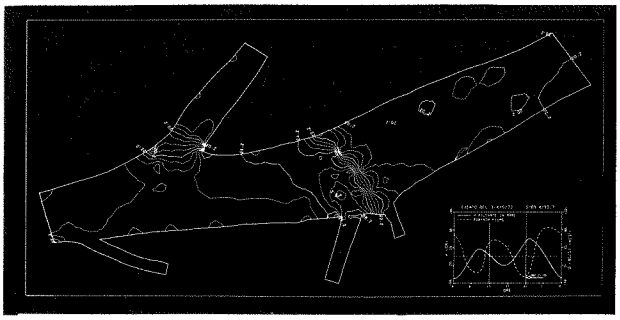


Fig. 12

Fig. 9

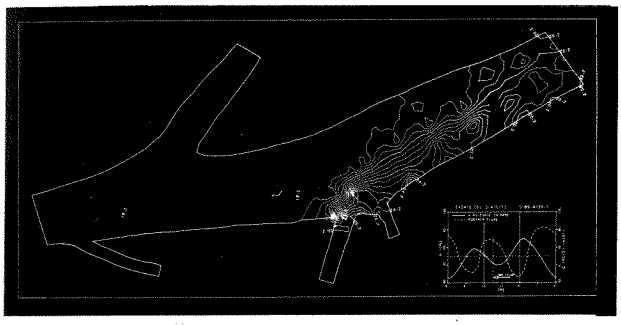


Fig. 13

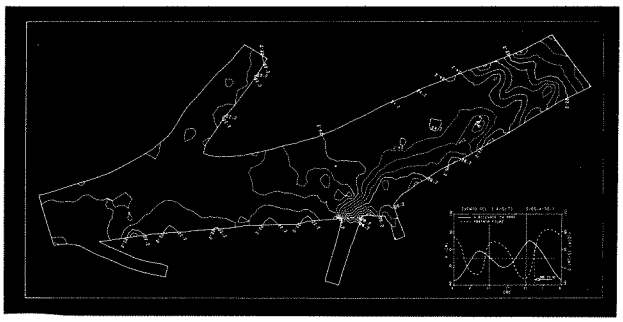


Fig. 10

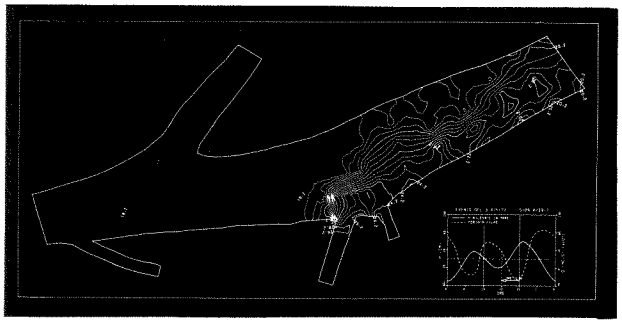


Fig. 14

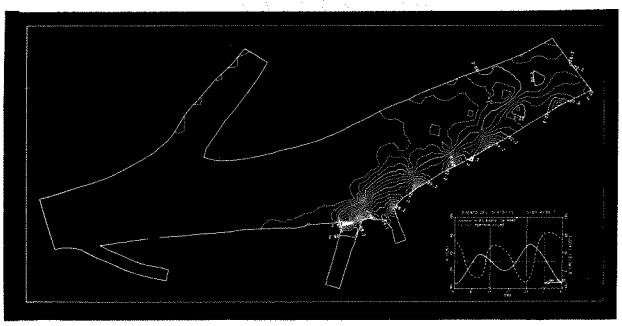


Fig. 11

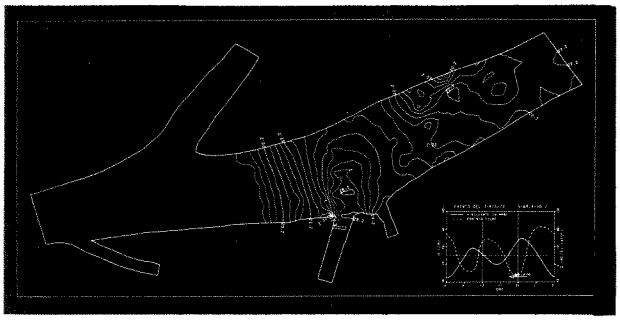
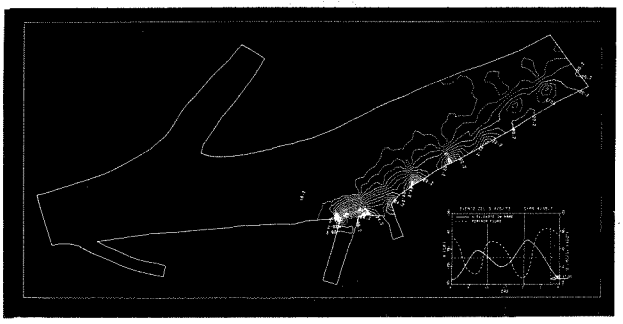


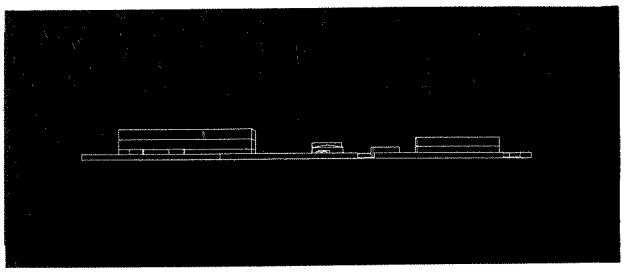
Fig. 15



dalla posizione dell'osservatore; poiché gli elementi che compongono l'insieme sono considerati opachi, automaticamente riconosce le linee ed i tratti di linea coperti alla vista e ne evita l'esibizione.
L'osservatore si muove in questo spazio seguendo una traiettoria inizialmente diritta in elevazione fino a raggiungere una distanza dagli edifici pari a 4 volte il loro sviluppo trasversale; quindi prosegue con moto circolare a quota costante, per allontanarsi tangenzialmente dopo una rotazione completa.
L'animazione è costituita da 1000 diverse prospettive ripetute ciascuna tre volte: quindi, in questo caso, al limite dei normali tempi di persistenza della immagine visiva. Proiettata alla velocità di 16 fotogrammi al secondo simula una velocità di spostamento dell'osservatore pari a 15 m/s.
Le figure 16 - 28 rappresentano viste successive ad intervalli regolari di tempo.
Altre animazioni attualmente in preparazione riguardano i moti ondosi in bacini, canali o vasche, il movimento di particelle in sospensione in fluidi in movimento, i modi di vibrare di strutture sottoposte ad urti.
Le possibilità di applicazione del metodo non hanno praticamente limite e la sintesi che esso è in grado di operare consente di rendere vivo, immediatamente e globalmente percepibile, ciò che è usualmente presentato in forma schematica e parziale.

Figg. 16-26 - Animazione della vista di un complesso industriale: le figure rappresentano i fotogrammi da diversi punti di osservazione.

Fig. 16



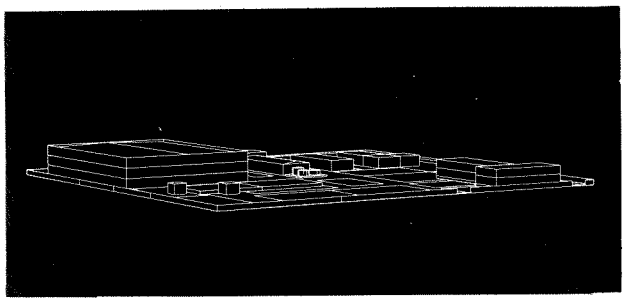


Fig. 17

Fig. 19

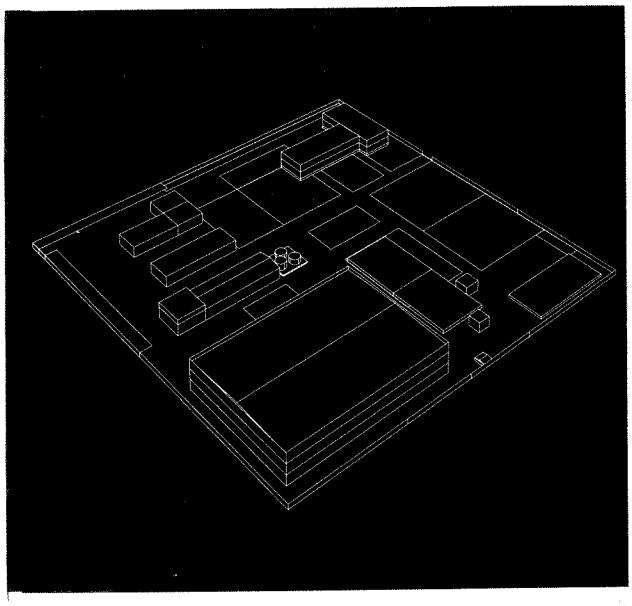


Fig. 18

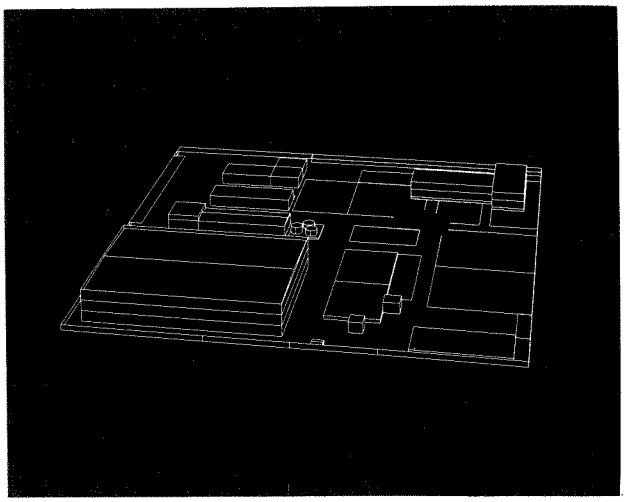
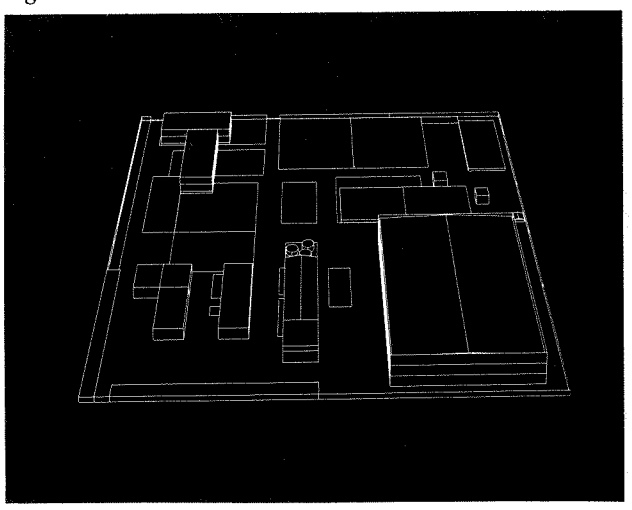


Fig. 20



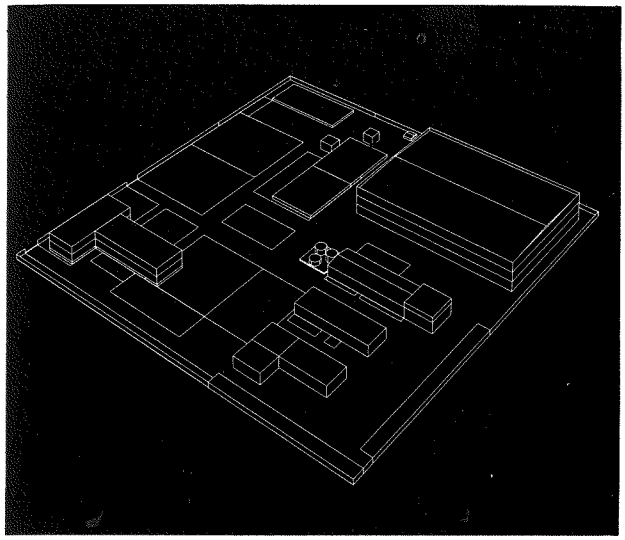


Fig. 21

Fig. 22

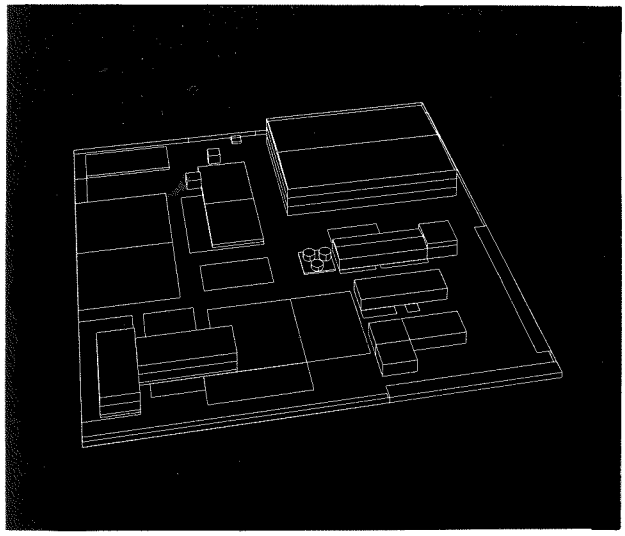
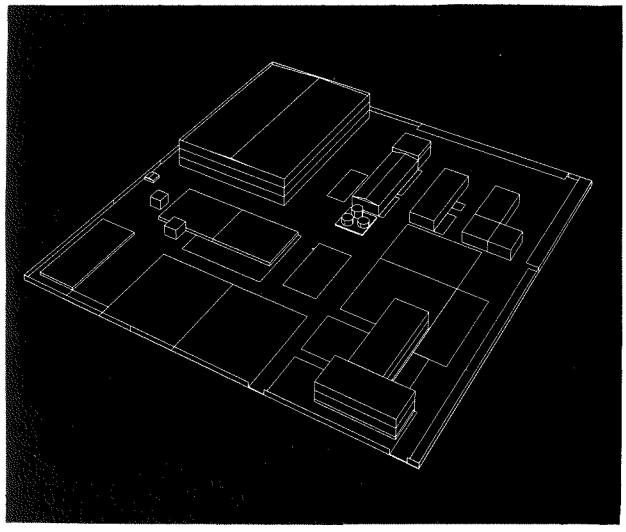


Fig. 23



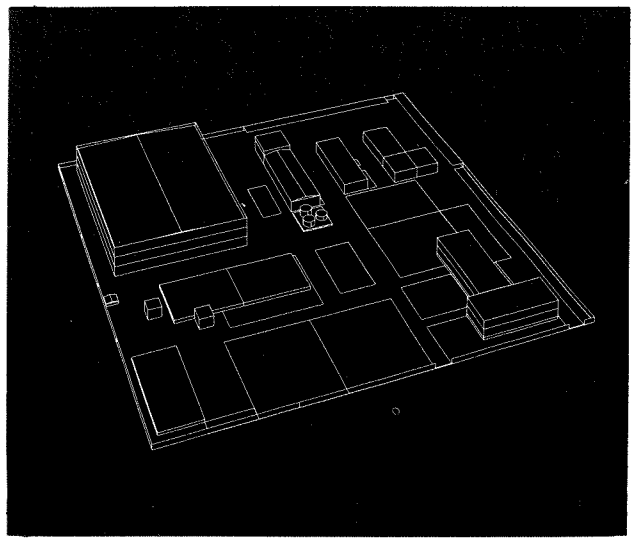


Fig. 24

Fig. 25

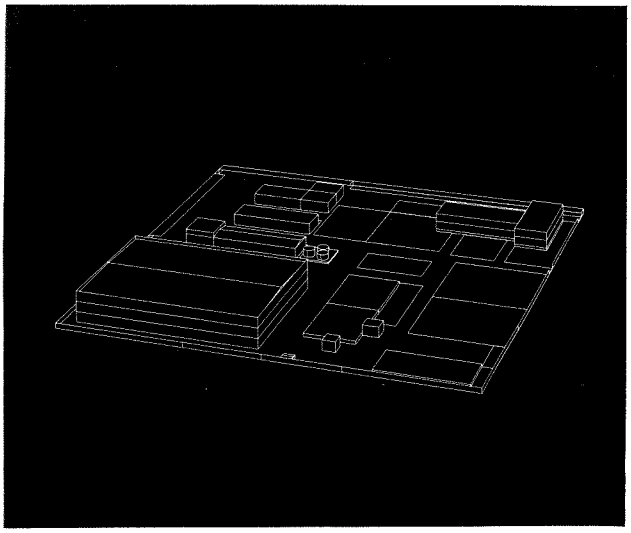
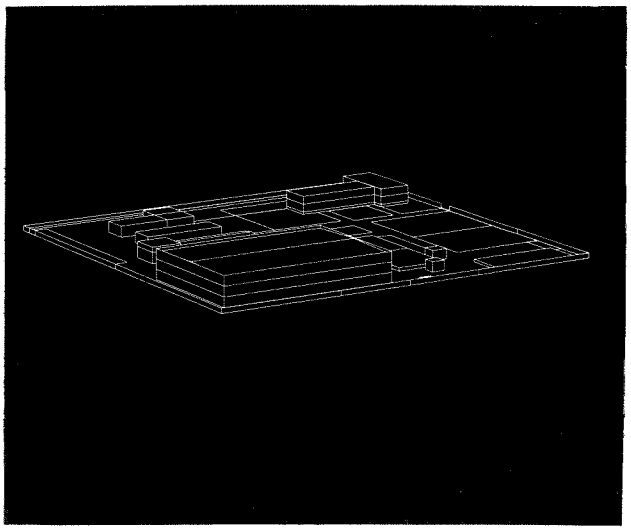


Fig. 26



L'informatica nelle amministrazioni comunali

ANTONIO BEZZOLA
Honeywell Information Systems Italia
Servizio Ricerche e Sviluppo Marketing

1 - Introduzione

Le Amministrazioni Comunali tendono a ricoprire un ruolo sempre più importante nell'apparato della Pubblica Amministrazione perchè, in particolare negli ultimi anni, sono state riconosciute a questi Enti competenze sempre più vaste su materie concernenti la gestione della popolazione e del territorio. A questo aumento delle competenze è seguito anche un aumento dei compiti da assolvere da parte delle Amministrazioni Comunali, aumento questo che, congiunto alla necessità di miglioramenti gestionali e di conoscenze della realtà, è stato un elemento che ha causato l'incremento della domanda di informatica in questo settore.
L'utilizzo della elaborazione elettronica dei dati permette infatti di assolvere vecchie e nuove competenze in maniera più efficiente, con possibili risparmi di personale, e di migliorare l'immagine esterna dell'apparato comunale.
In questo articolo si analizza il processo di diffusione della meccanizzazione nelle Amministrazioni Comunali; questo fenomeno è indagato dapprima confrontando i risultati ottenuti da una ricerca condotta nel 1973 (v. "Livello di meccanizzazione dei Comuni Italiani - Analisi e prospettive" Hisi - Servizio Studi di mercato, 1974) con quelli ottenuti da un'analoga ricerca svolta presso tutti i comuni meccanizzati al 31-12-78; i risultati di quest'ultima ricerca ci permetteranno inoltre di analizzare lo stato attuale della meccanizzazione, le principali esigenze applicative e le probabili tendenze del mercato nei prossimi anni.

2 - Linee evolutive - Il quinquennio 1973-1978

Le tendenze evolutive sono riportate nella tav. 1, nella quale sono messi a confronto il numero dei comuni meccanizzati in proprio nel 1973 e nel 1978. Nel quinquennio considerato l'utilizzo di elaboratori general-purpose è diventato da fenomeno sporadico o quantomeno concentrato nei comuni di grandi dimensioni un fenomeno molto diffuso non solo nei grandi comuni ma anche in quelli di medie dimensioni con una significativa presenza anche nei comuni di medio piccole dimensioni.
Questo fenomeno di diffusione della meccanizzazione è senza dubbio il frutto di parecchie cause che sinteticamente possono essere schematizzate nell'aumento dei volumi di informazioni da trattare, nei nuovi campi di intervento e nella necessità di ricorrere a elaboratori elettronici per ottenere un miglioramento gestionale dell'Ente.

- L'aumento dei volumi di informazioni da trattare è legato all'aumento generalizzato della popolazione residente nei comuni di grandi e medie dimensioni che ha caratterizzato lo sviluppo demografico italiano.
- I nuovi campi di intervento sono legati all'aumento delle materie di competenza comunale in seguito al decentramento indicati nella legge 382 del 22 luglio 1975 e nel successivo DPR 616. Anche in seguito all'approvazione della legge sanitaria questi campi di intervento vengono ulteriormente allargati.
- L'utilizzo di elaboratori elettronici per ottenere un miglioramento gestionale dell'ente è legato da una parte alla possibilità di riconvertire personale divenuto eccedente nelle aree meccanizzate a nuove funzioni e dall'altra alla possibilità di meccanizzazione dei lavori più ripetitivi alleggerendo in questo modo il lavoro dei dipendenti comunali.

A questo processo di diffusione è associato un analogo processo di ampliamento del parco potenziale (v. tav. 2) che registra un tasso di sviluppo medio anno espresso in punti del 21,9% passando globalmente da 515 mila punti ad oltre 1385 mila punti (tav. 2 e 3).
Questo elevato tasso di sviluppo è la risultante sia dei contributi delle nuove referenze sia dell'ampliamento dei sistemi installati. I comuni compresi tra i 30 e 100.000 abitanti, ove il contributo delle nuove referenze è stato massiccio, presentano tassi di sviluppo medi annui significativamente superiori a quelli dei comuni

Tav. 1 - COMUNI - Stato della meccanizzazione al 1973 e 1978

Popolazione (000)	Totale Comuni	Comuni meccanizzati '73	Comuni meccanizzati '78
> 300	11	8	10
100-300	36	16	24
50-100	63	13	35
30- 50	123	9	24
5- 30	1815	(*)	22
TOTALE	2048	46	115

(*) Nell'indagine non sono considerati i comuni con popolazione inferiore ai 30.000 abitanti; è comunque possibile stimare in 3 i comuni di questa classe meccanizzati.

Tav. 2 - COMUNI - Valore (*) potenziale parco installato al 1973 e 1978

Popolazione (000)	Valore parco potenziale'73 (000)	Valore parco potenziale'78 (000)	Tasso di sviluppo medio annuo %
> 300	306,3	610,5	14,8
100-300	153,9	372,3	19,3
50-100	37,7	274,5	48,7
30- 50	17,7	86,2	37,2
5- 30	/	42,2	/
TOTALE	515,6	1385,7	21,9

(*) Il valore del parco è espresso in punti internazionali pari al canone di locazione mensile in dollari.

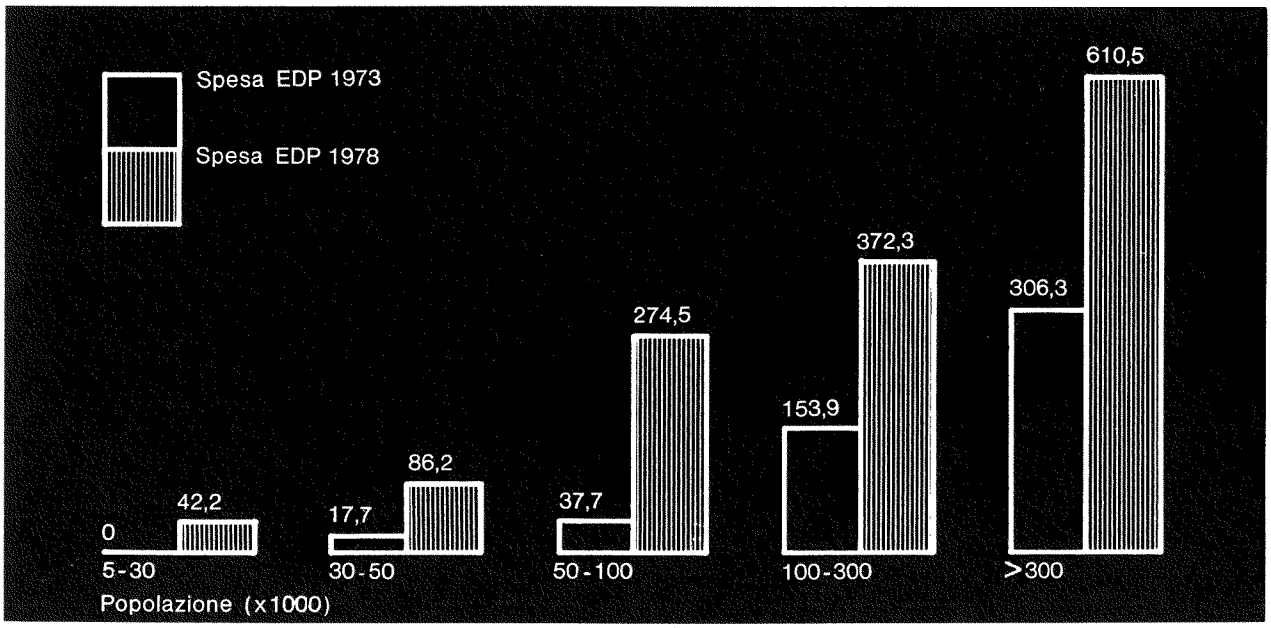
con popolazione superiore ai 100.000 abitanti, comuni per i quali il contributo delle nuove referenze è più basso.

3 - Stato della meccanizzazione

In seguito al processo di crescita descritto nel paragrafo precedente la situazione della meccanizzazione dei Comuni italiani al 31-12-78 ed analizzata secondo differenti classi dimensionali è quella riportata nella tav. 4. Il tasso di penetrazione EDP tende, sia a livello nazionale che all'interno di ciascuna area considerata, a diminuire col diminuire della popolazione residente nel comune.
Questo rapporto di proporzionalità diretta tra popolazioni e tasso di penetrazione è senza dubbio spiegato dal fatto che, ove si consideri la popolazione un indicatore dei volumi di lavoro che il comune deve svolgere, al di sopra di certe dimensioni demografiche, è ormai necessario ricorrere all'ausilio di un elaboratore elettronico.

Una significativa presenza della meccanizzazione è riscontrabile nei comuni con popolazione superiore ai 40.000 abitanti ed in particolare ciò è vero per l'Italia settentrionale e centrale.
La disaggregazione territoriale mette inoltre in evidenza una relativa arretratezza del sud rispetto alle altre due aree perchè il tasso di penetrazione EDP ha valori significativamente inferiori in quest'area.
Questa situazione è il riflesso dell'arretratezza economica di quest'area e di ormai consolidate prassi amministrative.
Alla luce di queste osservazioni è possibile prevedere nei prossimi anni uno sviluppo della meccanizzazione anche nella fascia di comuni con popolazione compresa tra i 20 e 40.000 abitanti soprattutto ove si consideri che il tasso di penetrazione in queste fascie dell'Italia Settentrionale, area pilota, è già del 15,7%.
La dimensione media dei sistemi installati per classe dimensionale dei comuni riportati nella tav. 4 tende a crescere col crescere delle dimensioni dell'Ente.
La dimensione del sistemi è evidentemente correlata

Tav. 3 - Dinamica spesa EDP



alla mole di lavoro da svolgere; è stata così calcolata la retta di regressione tra dimensione del sistema, espressa in punti, e volume di lavoro, espressa in numero di abitanti.
La retta così calcolata è la seguente:

$$Y = .0519\ X + 3815{,}2385 \qquad R^2 = .7514$$

Y = valore sistemi

X = popolazione residente nel comune

Il valore del coefficiente di determinazione (R^2), pur indicando che la dimensione del sistema è evidentemente correlata alla popolazione residente, indica anche che esiste una elevata variabilità del punteggio per abitanti.
Esistono inoltre soluzioni differenti dalla costituzione di un CED proprio per effettuare la meccanizzazione del comune.
Le soluzioni più diffuse per il numero di comuni coinvolti, è quella rappresentata dalla costituzione di un unico CED a carattere regionale che serve tutti i principali enti locali di quell'area: comuni, province, regione, camere di commercio, ospedali, centri di ricerca regionale ecc.
Questa soluzione è a tutt'oggi presente in 3 regioni italiane: Friuli Venezia Giulia, Umbria e Basilicata.
Queste soluzioni, che complessivamente prestano servizi a 13 comuni, paiono essere adatte solo per regioni di dimensioni limitate, come le tre prima citate, perchè, per regioni di dimensioni maggiori, i problemi derivanti dai volumi di lavoro rendono poco economica questa soluzione.
Altre due soluzioni sono rappresentate dall'appoggio a Centri servizi privati o della costituzione di CED consortili.
L'appoggio a Centri servizi privati per una gestione completa del comune è una soluzione riscontrata solo in rari casi.
Ugualmente rara è la soluzione consortile che però pare destinata ad avere maggior peso perchè può risolvere il problema della meccanizzazione nei comuni di piccole dimensioni (anche comuni con popolazione inferiore ai 20.000 abitanti). Per questi comuni infatti l'onere di gestione del centro ed anche il reperimento di personale qualificato costituisce un problema di non facile soluzione, ma facilmente risolvibile se affrontato unitamente ad altri comuni.
A questa soluzione si sono però opposti, quando si è cercato di intraprendere questa via, difficoltà di carattere politico e tradizioni municipalistiche.

4 - Esigenze applicative

Le esigenze applicative delle Amministrazioni Comunali sono facilmente identificabili, perchè investono precise aree.
Le aree a tutt'oggi meccanizzate (v. tav. 5) sono principalmente quella demografica e finanziaria, includendo in quest'ultima anche la gestione del personale; sono inoltre meccanizzate saltuariamente altre funzioni di competenza comunale quali le contravvenzioni, la tenuta dei registri dei pesi e delle misure o l'effettuazione della bollettazione per le aziende municipalizzate.

- L'AREA DEMOGRAFICA comprende la gestione dell'archivio anagrafico e le relative certificazioni anagrafiche, procedura che nella maggior parte dei casi è gestita in TP, la gestione degli archivi elettorali con le revisioni e la stampa dei certificati in occasione delle elezioni, la determinazione delle liste di cittadini che appartengono alle leve militari e scolastiche e la gestione di un archivio vaccinale che prevede alla chiamata dei bambini da vaccinare, e, in alcuni casi la certificazione in TP delle vaccinazioni effettuate. Pur essendo queste quattro procedure collegate tra di loro perchè tutte fanno riferimento ad un unico archivio della popolazione esse hanno una frequenza di meccanizzazione differenziata ad indicare che la meccanizzazione di una procedura non implica l'automazione di tutte le altre procedure appartenenti alla stessa area. È stato anche riscontrato che l'integrazione non è stata sempre effettuata perchè la meccanizzazione delle diverse procedure è avvenuta in tempi differenti senza aver effettuato una programmazione del lavoro di sviluppo del Centro.
La meccanizzazione dell'area demografica oltre a permettere di svolgere le funzioni di competenze comunali quali la certificazione anagrafica o l'emissione delle

Tav. 4 - COMUNI: Tasso di penetrazione per area geografica, classe dimensionale e dimensione media dell'utente EDP

Popolazione (000)	Tasso di penetrazione EDP				Dimensione
	Nord	Centro	Sud	Tot. Italia	Media utente in punti (000)
> 500	100,0	100,0	100,0	100,0	85,0
200-500	83,3	100,0	80,0	83,3	22,2
70-200	71,4	69,2	35,3	60,3	12,0
40- 70	51,7	44,4	15,8	35,1	5,1
20- 40	15,7	3,7	6,4	9,4	2,2
5- 20	0,7	0,3	0,4	0,5	1,6
TOTALE	6,6	6,7	3,6	5,6	12,1

liste di leva permette anche di avere la situazione aggiornata dello stato della popolazione fornendo così un utile strumento per le scelte di carattere economico, politico e sociale che gli Amministratori devono prendere. È ad esempio possibile ottenere dati sui flussi migratori interni alla città, sulla età della popolazione, sull'ammontare esatto della popolazione residente e qualsiasi altra indicazione reperibile nei dati costituenti l'archivio.

Per i comuni di grandi dimensioni la meccanizzazione di queste procedure permette il decentramento degli uffici di certificazione nei quartieri, altrimenti impossibile per lo spazio occorrente alla tenuta di un targhettario, offrendo così un indubbio servizio a tutta la cittadinanza.

• L'AREA FINANZIARIA che si articola nella gestione del bilancio, nelle determinazioni delle paghe e degli stipendi e nelle gestioni di ruoli delle imposte comunali costituisce l'area di più frequente meccanizzazione.

È infatti la procedura della determinazione delle paghe e stipendi quella con frequenza di meccanizzazione più alta in assoluto.

La gestione del bilancio prevede la completa gestione di tutti i movimenti finanziari che si riferiscono sia al bilancio di competenza che ai residui. Le sue funzionalità coprono tutte le aree di competenza della ragioneria generale.

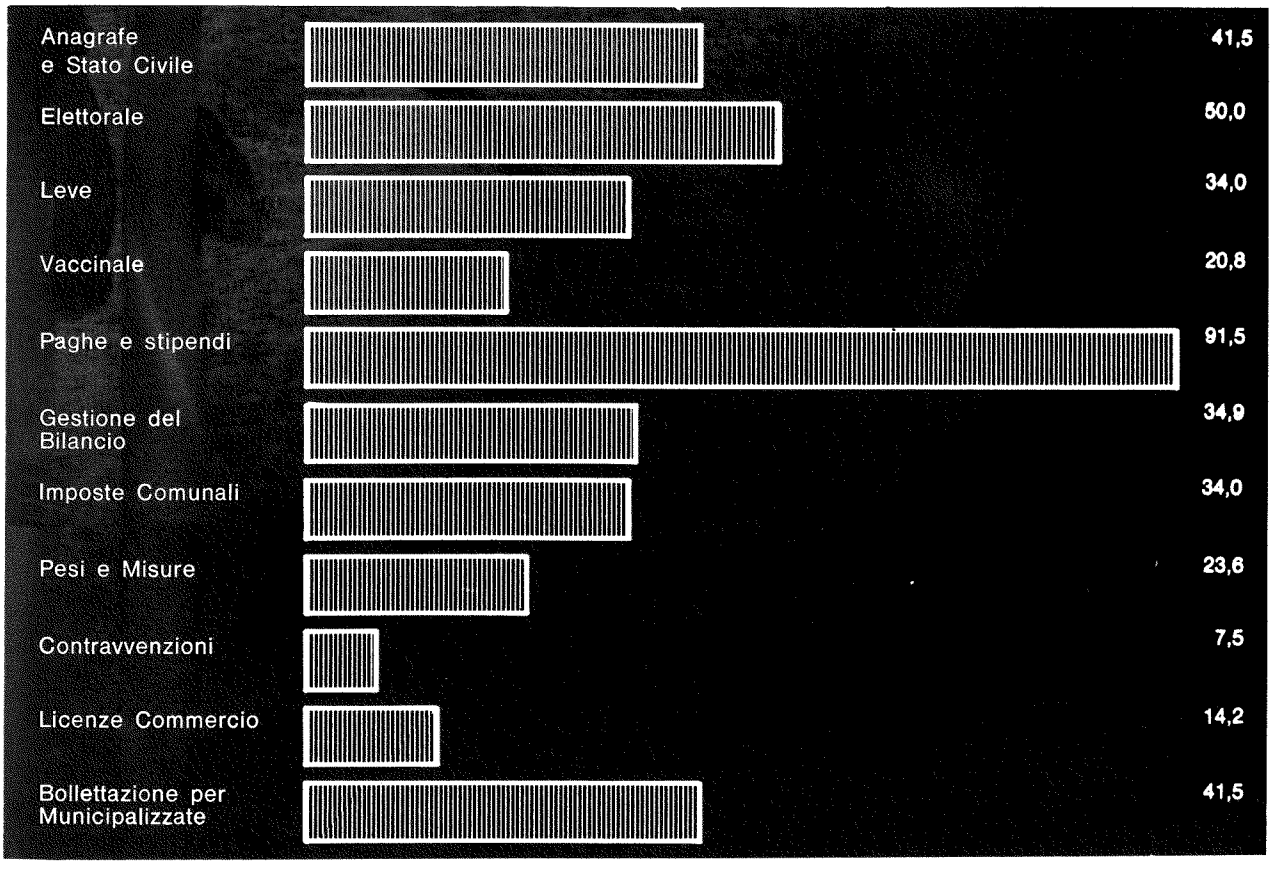
La meccanizzazione di questa procedura non è molto elevata per la complessività della procedura stessa e per le resistenze interne dell'apparato burocratico comunale. Nel caso di meccanizzazione di questa procedura gli amministratori hanno a propria disposizione in qualsiasi momento, soprattutto se è stata realizzata la gestione TP, lo stato di avanzamento del bilancio per ciascun capitolo permettendo così concreti miglioramenti gestionali.

La gestione dei ruoli delle imposte comunali (tasse sui cani, sulle affissioni, sulla rimozione dei rifiuti solidi e sulla occupazione di spazi pubblici) è una procedura con scarsa frequenza di meccanizzazione e si limita a gestire un archivio in cui sono registrati tutti i ruoli delle persone che devono pagare queste imposte.

Le altre procedure meccanizzate, fatta eccezione per la bollettazione delle aziende municipalizzate hanno una frequenza di meccanizzazione molto bassa. In particolare le contravvenzioni sono meccanizzate solo dai comuni di grandi dimensioni. Esistono inoltre casi singoli di automazione di altre aree quali quella della gestione, sia in fase preventiva che di contabilizzazione, delle opere pubbliche. Per la sporadicità dei casi, non sono queste le aree che a tutt'oggi possono essere considerate di primario interesse da parte della amministrazione comunale.

Un'ultima osservazione deve essere fatta a proposito dello sviluppo del software applicativo. L'affidamento dei CED a società di consulenza per lo sviluppo delle procedure nelle fasi di avvio è riscontrata solo nei co-

Tav. 5 - Frequenza di meccanizzazione delle singole procedure



muni di grandi dimensioni. Per i comuni di medio e medio piccole dimensioni il CED è gestito in proprio e, almeno fino al 1976, nella quasi totalità le procedure erano sviluppate in proprio andando così incontro a tempi molto lunghi di avvio.
Negli ultimi due anni invece la situazione tende ad invertirsi; tutti i comuni meccanizzati dopo il 76 hanno utilizzato procedure sviluppate o dalle case fornitrici di hardware o da altri comuni o Enti Locali. Ciò è accaduto sia per una aumentata sensibilità delle case fornitrici di hardware sia per una maggior sensibilità dei comuni al problema dei pacchetti applicativi.
Si è voluto a questo proposito condurre un'analisi più attenta su un campione di 25 comuni stratificato secondo le dimensioni e la distribuzione geografica per avere indicazioni più precise sull'utilizzo dei CED.
Come illustrato nella tav. 6 che riporta il numero di procedure in carico al CED, risulta che oltre ad esservi tre comuni che non utilizzano assolutamente il CED, parecchi comuni utilizzano il CED molto poco.
È stato così calcolato un indice in grado di esprimere l'utilizzo del CED. Questo indice è stato calcolato assegnando un punteggio ad ogni procedura (1 per le procedure periodiche; 6 per le procedure giornaliere e 3 per le procedure per le municipalizzate) e confrontando il valore così ottenuto, esprimente gli impieghi del CED, con le risorse disponibili misurate in punti per migliaia di abitanti, tenendo così conto delle dimensioni del comune.
Analiticamente l'indice è stato così ricavato:

$$\text{indice di utilizzo} = p/s$$

dove p è il valore del sistema diviso per il numero degli abitanti ed s è la somma dei punteggi delle procedure in carico al CED.
I risultati, riportati nella tav. 7, confermano e qualificano un giudizio negativo sullo stato di meccanizzazione dei comuni.
Come si è già precedentemente accennato, questo scarso utilizzo è dovuto nella quasi totalità dei casi alle difficoltà che qualsiasi utente di elaboratori elettronici si trova ad affrontare intraprendendo per la prima volta la meccanizzazione. I tempi di avvio, in mancanza di pacchetti applicativi già predisposti delle case fornitrici di hardware o da società di consulenze si rivelano estremamente lunghi soprattutto quando, come nei casi dei comuni, esistono delle difficoltà a reperire tecnici esperti.
Una soluzione a questo problema, che si riflette anche sulla redditività dell'investimento in informatica, è senz'altro l'adozione di pacchetti applicativi studiati per la realtà comunale ed in grado di permettere un'avviamento del CED in tempi brevi ed uno standard di prestazioni elevato.

5 - Le prospettive

Sulla base di tutte le osservazioni precedentemente fatte è possibile effettuare qualche previsione sulle prospettive della meccanizzazione dell'apparato comunale:

Tav. 6 - Suddivisione dei comuni in base al numero di procedure in carico al CED

Numero procedure in carico al CED	Numero Comuni
0	3
1	3
2	3
3	1
4	4
5	6
6	2
7	/
8	3

- dal punto di vista quantitativo è possibile prevedere da una parte una crescita del tasso di penetrazione arrivando così nel giro dei prossimi anni, ad una quasi completa meccanizzazione dei comuni con popolazione superiore ai 40.000 abitanti ed una massiccia diffusione nella fascia di comuni compresa tra i 20 e 40.000 abitanti.
Nel giro di due-tre anni non pare però che il Mezzogiorno d'Italia possa recuperare del tutto il gap accumulato nei precedenti 5 anni.
Al presumibile aumento della qualità e della quantità dei servizi prestati dal comune alla cittadinanza dovrà fare evidentemente riscontro un aumento della dimensione media del sistema installato e questo aumento sarà molto marcato nei comuni di medio e medio piccole dimensioni (40.000 - 200.000 abitanti).

- È anche possibile che nei prossimi anni faccia la propria comparsa nelle amministrazioni comunali l'informatica distribuita soprattutto nei grandi comuni o in soluzioni consortili. La possibilità infatti di utilizzare della capacità di elaborazione decentrata e della trasmissione al centro di dati già elaborati è senz'altro una soluzione di elevato interesse per le classi di utenti prima elencati.

- A fronte anche di una maggiore sensibilità delle case costruttrici di hardware e di un coordinamento delle autorità è possibile prevedere, e già in parte se ne vedono i sintomi, una tendenza alla omogeneità delle procedure adottate permettendo così un aumento della redditività dell'investimento in informatica, evitando spese di sviluppo di procedure e tempi morti per l'avviamento del centro. Si è d'altronde precedente-

Tav. 7 - Numero di Comuni per indice di utilizzo

Indice di utilizzo	Numero Comuni
$\leqslant 10$	11
10-20	4
20-30	4
30-40	4
> 40	2

mente visto come questa tendenza sia, almeno in parte già in atto.

• Per quanto riguarda l'aspetto applicativo i nuovi campi di sviluppo sono evidentemente legati a quella che sarà la produzione giuridica sul decentramento. A tutt'oggi è già possibile prevedere quale futuro campo di sviluppo, la gestione patrimoniale perchè verranno compresi nel patrimonio comunale i beni ora di proprietà della IPAB e degli Ospedali. La possibile introduzione della contabilità di cassa e l'obbligo di informare trimestralmente le autorità di controllo sullo stato del bilancio di cassa sarà un impulso alla meccanizzazione della gestione del bilancio perchè solo con una gestione elettronica sarà possibile, a fronte di certi volumi, effettuare queste segnalazioni.

Un'altra area, non ancora meccanizzata ma già in fase di studio in alcuni comuni, è la gestione del territorio.

Con questo termine si intende la gestione meccanizzata del catasto, del piano commerciale comunale, della concessione delle licenze edilizie, della riscossione delle tasse sulla rimozione dei rifiuti solidi, delle tasse dei passi carrai e della registrazione degli scarichi industriali nelle acque del comune. È questa un'area molto vasta ma di particolare interesse anche se la sua realizzazione appare molto complessa.

Altri campi di applicazione, di cui però non è possibile entrare nello specifico per mancanza di legislazione definitiva, riguardano la riforma sanitaria (legge 833 del 23 dicembre 1978) e la costituzione delle unità sanitarie locali.